Igor Ushakov

OPTIMAL RESOURCES ALLOCATION

AUTHOR’S PREFACE

In Memory of John D. Kettelle Jr.
my friend, colleague and informal teacher

This book is in memory of my friend and colleagues Dr. John D. Kettelle, a former mariner who fought in WWII and later made a significant input in dynamic programming. His name was known to me in late 1960-s when I was a young engineer in the former Soviet Union. I had been working at one of the R&D institutes of the Soviet military-industrial establishment; my duty was projecting spare stocks for large scale military systems.

I met Dr. J. Kettelle in person in early 1990-s when I came to the United States as Distinguished Visiting Professor at The George Washington University. After two years at the University, I was invited by John to work at Ketron, Inc., the company that was established and led by him. We became friends.

I will remember John forever…

* * *

Optimal resource allocation is an extremely important part of many human activities, including reliability engineering. One of the first problem arose in this engineering area was optimal allocation of spare units. Then it came to optimization of networks of various natures (communication, transportation, energy transmission, etc.) and now it is an important part of counter-terrorism protection.

Actually, these questions always stood and stand: How to achieve maximum gain with limited expenses? How to fulfill requirements with minimum expenses?

In this book, one finds an overview of different approaches of optimal resource allocation, from classical LaGrange methods to modern heuristic algorithms.

This book is not a tutorial in a common sense of words. It is not a reliability “cooking book”. It is sooner a bridge between reliability engineering and applied mathematics in the field of optimal allocation of resources for systems’ reliability increase. It supplies the reader with basic knowledge in optimization theory and presents examples of application of the corresponding mathematical methods to the Real World problems. The book objective is to inspire the reader visiting the wonderful area of applied methods of optimization, rather than give them a mathematical course on optimization.

Examples with sometimes tedious and bulky numerical calculations should not frighten the Reader. They are given with the only purpose: to demonstrate “a kitchen” of calculations. All these calculations have to be performed by a computer. Optimization programs themselves are enough simple. (For instance all numerical examples were performed with the help a simple program in MS Office Excel.)

In the very end of the book there is a complete enough list of monographs on the topic.

* * *

Who are potential readers of the book? First of all, engineers who design complex systems and mathematicians who are involved in “mathematical support” of engineering projects. Another wide category is college and university students, especially, before they take classes on optimization theory. At last, university professors could use the material in the book taking numerical examples and case studies for illustration of the methods they are teaching.

* * *

In conclusion, I would like to say a few words about references at the end of chapters. Actually, each of them is not a list of reference, but rather a bibliography presented in a chronological order. The author’s belief is that such list will allow the reader to trace the"evolution" of the considered topic. The lists, of course, are not full, for which the author in advance brings his apology. However, as Kozma Prutkov (a pseudonym for the group of pokemon satirists the end of the 19th century) said: "Nobody can embrace the unembraceable".

Igor Ushakov

San Diego, USA, 2012

* * *

Table of Contents

The problem of optimal resources allocation for antiterrorism preventive measures is naturally based on subjective estimates made by experts in this field.

1. BASIC MATHEMATICAL REDUNDANCY MODELS

A series system of independent subsystems is usually considered as a starting point for optimal redundancy problems. The most common case is when one considers a group of redundant units as a subsystem. The *reliability objective function of a series system* is usually expressed as a product of probabilities of successful operation of its subsystems. The *cost objective function is usually assumed a linear function of the number of system's units.*

There are also more complex models (multi-purpose systems and multi-constrain problems) or more complex objective functions like average performance or the mean time to failure. However, we don't limit ourselves with pure reliability models. The reader will find a number of examples with various networks as well as examples of resource allocation in counter-terrorism protection.

In the book we consider main practical cases, describe various methods of solutions of optimal redundancy problems, and demonstrate solving of the problems with numerical examples. Finally, several case studies are presented that reflect the author's personal experience and can demonstrate practical applications of presented applied methodology.

1.1. *Types of Models*

A number of various mathematical models of systems with redundancy have been developed during about half a century of developing modern reliability theory. Some of these models are rather specific and some of them are even "extravagant". We limit ourselves in this discussion to the main types of redundancy and demonstrate on them how methods of optimal redundancy can be applied to solutions of the optimal resource allocation.

Redundancy in general is a wide concept, however, we mainly will consider the use of a redundant unit to provide (or increase) system reliability.

Let us call a set of operating and redundant units of the same type *a redundant group*. Redundant units within a redundant group can be in one of the two states: active (in the same regime as operating units, i.e. so-called "hot redundancy") and standby (idle redundant units waiting to replace failed units, i.e. so-called "cold redundancy").

In both cases there are two possible situations: failed units could be repaired and returned to the redundant group or unit failures lead to exhaustion of the redundancy.

In accordance with such very rough classifications of redundancy methods, this chapter structure will be arranged in the following way:

Table 1.1. Types of redundancy

		Redundant units regime	
		Active	Standby
Type of maintenance	Non-repairable	Section 1.1	Section 1.2
	Repairable	Section 1.3	Section 1.4

We consider two main reliability indices: probability of failure-free operation during some required fixed time t_0, $R(t_0)$, and mean time to failure, T.

In practice, we often deal with a system consisting of a serial connection of redundant groups:

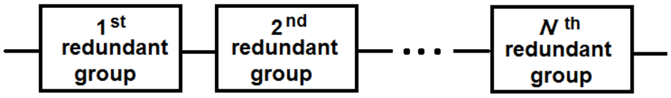


Figure 1.1. General block diagram of series connection of redundant groups.

Usually, such kind of structures is found in systems with spare stocks with periodical replenishment.

1.2. Non-repairable redundant group with active redundant units

Let us begin with a simplest redundant group of two units (duplication).

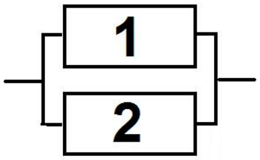


Figure 1.2. Block diagram of a duplicated system.

Such system operates successfully if at least one unit is operating. If one denotes random time to failure of unit k by ξ_k , then the system time to failure, ξ, could be written as

$$\xi = \max \{\xi_1 , \xi_2\} \tag{1.1}$$

The following time diagram explains equation (1.1):

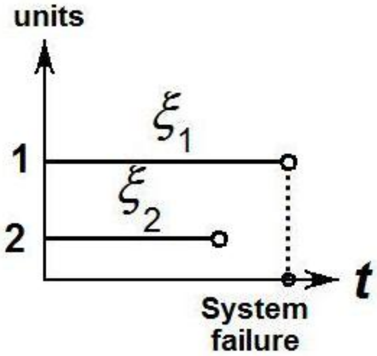


Figure 1.3. Time diagram for a non-repairable duplicated system with both units active

The probability of failure-free operation (PFFO) during time t for this system is equal to

$$R(t) = 1– [1-r(t)]^2 \qquad (1.2)$$

where $r(t)$ is PFFO of a single active unit.

We will assume an exponential distribution of time to failure for an active unit:

$$F(t) = \exp(-\lambda t). \qquad (1.3)$$

In this case the mean time to failure (MTTF), T , is equal to:

$$T = E\{\xi\} = E\{\max(\xi_1, \xi_2)\} = \int_0^\infty R(t)dt = \int_0^\infty 1 - [1 - \exp(\lambda t)]^2 \, dt = (1 + 0.5) \cdot \frac{1}{\lambda} \qquad (1.4)$$

Now consider a group of n redundant units that survives if at least one unit is operating.

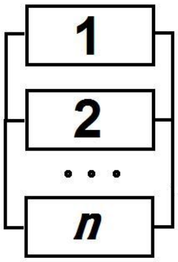


Figure 1.4. Block diagram of redundant group of *n* active units.

We omit further detailed explanations that could be found in any textbook on reliability (Bibliography to Chapter 1).

For this case PFFO is equal:

$$R(t) = 1– [1-r(t)]^n \qquad (1.5)$$

and the mean time to failure (under assumption of the exponential failure distribution) is

$$T = \sum_{1 \le k \le n} \frac{1}{k}. \qquad (1.6\ )$$

The most practical system of interest is the so-called “*k* out of *n*” structure. In this case, the system consists of *n* active units in total. The system is deemed to be operating successfully if *k* or more units have not failed. (Sometimes this type of redundancy is called “floating”). The simplest system frequently found in engineering practice is a “2 out of 3” structure.

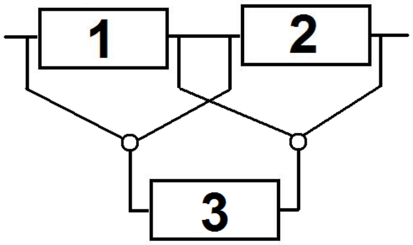


Figure 1.5. Block diagram of a “2 out of 3” structure with active redundant unit.

A block diagram for general case can be presented in the following conditional way. It is assumed that any redundant unit can immediately operate instead of any of *k* “main” units in case a failure.

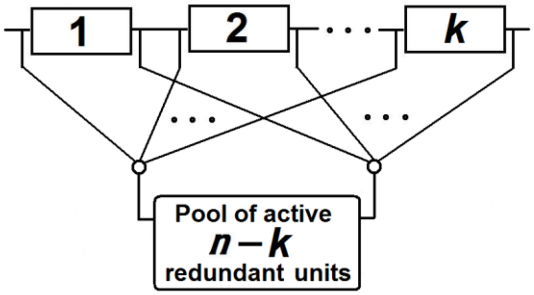


Figure 1.6. Block diagram of a “*k* out of *n*” structure with active redundant units.

Redundancy of this type can be found in multi-channel systems, for instance, in base stations of various telecommunication networks: transmitter or receiver modules form a redundant group that includes operating units as well as a pool of active redundant units.

Such system is operating until at least *k* of its units are operating (i.e. less than *n* – *k* + 1 failures have occurred). Thus, PFFO in this case is

$$R(t) = \sum_{k \le j \le n} \binom{n}{j} [p(t)]^j [1 - p(t)]^{n-j} \tag{1.7}$$

and

$$T = \frac{1}{\lambda} \sum_{k \le j \le n} \frac{j}{n}. \tag{1.8}$$

If a system is highly reliable, sometimes it is more reasonable to use (1.7) in supplementary form (especially, for approximate calculations when *p*(*t*) is close to 1).

$$R(t) = 1 - \sum_{n-k+1 \le j \le n} \binom{n}{j} [1 - p(t)]^j [p(t)]^{n-j} \approx 1 - \binom{n}{n-k+1} [1 - p(t)]^{n-k+1} \tag{1.9}$$

1.3. Non-repairable redundant group with standby redundant units

Again begin with a duplicated system presented on the following figure.

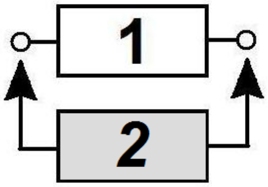


Figure 1.7. A non-repairable duplicated system with a standby redundant unit. (Here grey color denotes a standby unit.)

For this type of systems, the random time to failure is equal to:

$$\xi = \xi_1 + \xi_2 \quad . \qquad (1.10)$$

The following time diagram explains equation (1):

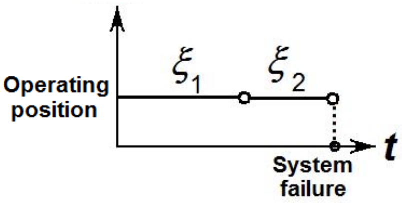


Figure 1.8. Time diagram for a non-repairable duplicated system with a standby redundant unit.

The PFFO of a considered duplicate system can be written in the form:

$$R(t) = p_0(t) + p_1(t) \qquad (1.11)$$

where $p_0(t)$ is the probability of no failures at time interval [0, t], and $p_1(t)$ is the probability of exactly one failure in the same time interval. Under assumption of exponentiality of the time-to-failure distribution, one can write:

$$p_0 = \exp(-\lambda t) \qquad (1.12)$$

and

$$p_1 = \lambda t \exp(-\lambda t)\,, \qquad (1.13)$$

so finally

$$R(t) = \exp(-\lambda t) \cdot (1 + \lambda t)\,. \qquad (1.14)$$

Mean time to failure is defined as

$$T = E\{\xi_1 + \xi_2\} = \frac{2}{\lambda}\,, \qquad (1.15)$$

since λ=1/T.

For a multiple standby redundancy a block diagram can be presented in the following form:

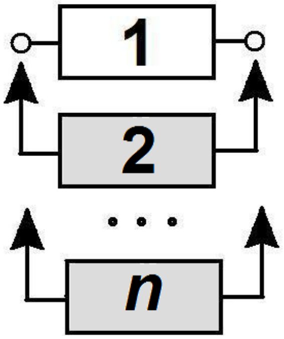


Figure 1.9. Block diagram of redundant group of one active and *n* – 1 standby units. (Here grey boxes indicate standby units.)

For this redundant group, one can easily write (using the arguments given above):

$$R(t) = \exp(-\lambda t) \sum_{1 \le j \le n-1} \frac{(\lambda t)^j}{j!} \tag{1.16}$$

and

$$T = \frac{n}{\lambda} \tag{1.17}$$

A block diagram for a general case of standby redundancy of *k* out of *n* type can be presented in the following way.

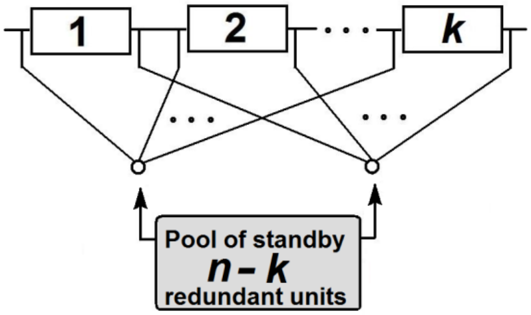


Figure 1.10. Block diagram of a "*k* out of *n*" structure with standby redundant units. (Here grey color is used to show standby redundant units).

It is assumed that any failed operational unit can be instantaneously replaced by a spare unit. Of course, no replacement can be done instantaneously: speaking so we keep in mind a five seconds rule[1] ☺.

This type of redundant group can be found in spare inventory with periodical restocking. Such replenishment is typical, for instance, for terrestrially distributed base stations of global satellite telecommunication systems. One observes a Poisson process of operating units failures with parameter *kλ*, and the group operates until the number of failures exceeds n – k. The system PFFO during time *t* is equal to:

[1] Russian joke: If fallen object is picked up in 5 seconds , it is assumed as non-fallen at all.

$$R(t) = \exp(-k\lambda t) \cdot \sum_{0 \le j \le n-k} \frac{(k\lambda t)^j}{j!} \tag{1.18}$$

and the system MTTF is

$$T = \frac{1}{\lambda} \cdot \frac{n-k+1}{k} \tag{1.19}$$

Remark. Of course, there are more complex structures that involve active and standby redundant units within the same redundant group. For instance, structure "*k* out of *n*" with active units could have additional "cold" redundancy that allows performing "painless" replacements of failed units.

1.4. Repairable redundant group with active redundant units

Consider a group of two active redundant units, i.e. two units in parallel. Each unit operates independently: after failure it is repaired during some time and then returns to its position. Behavior of each unit can be described as alternating stochastic process: a unit change its states: one of proper working that during time ξ, followed by a failure state induced repair interval, η. The cycle of working/repairing repeats. This process is illustrated in Figure 11.

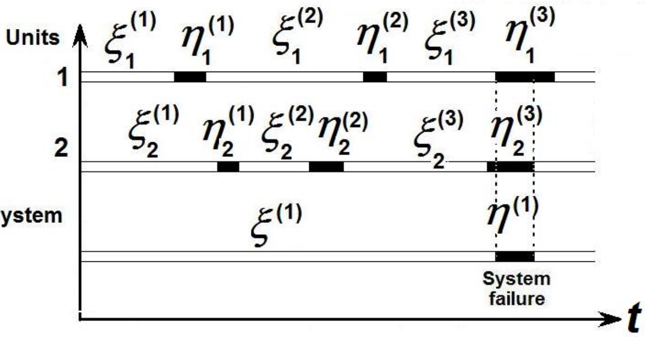


Figure 1.11. Time diagram for a repairable system with standby redundancy. White parts of a strip denote operating state of a unit and black parts do its failure state. Here $\xi_j^{(i)}$ denotes *j*-th operating interval of unit *i*, and $\eta_j^{(i)}$ does *j*-th interval of repair of this unit.

From Figure 1.11, one can see that the system failure occurs when failure intervals of both units overlapped.

Notice that for repairable systems, one of the most significant reliability indices is the so-called availability *coefficient,* $\tilde{r}$. This reliability index is defined as the probability that the system is in a working state at some arbitrary moment of time. (This moment of time is assumed to be "far enough" from the moment of the process start.) It is clear that this probability for a single unit is equal to a portion of total time when a unit is in a working state, i.e.

$$\tilde{r} = \frac{E\{\xi\}}{E\{\xi\}+E\{\eta\}.} \tag{1.20}$$

If there are no restrictions, i.e. each unit can be repaired independently, the system availability coefficient, $\tilde{R}$, can be written easily

$$\tilde{R} = 1-(1-r)^2 \tag{1.21}$$

For general types of distributions, reliability analysis is not simple. However, if one assumes exponential distributions for both ξ and η, reliability analysis can be performed with the help of Markov models.

If redundant group consists of two units, there are two possible regimes of repair, depending on the number of repair facilities. If there is a single repair facility, units become dependent through the repair process: the failed unit can find the facility busy with the repair of a previously failed unit. Otherwise, units operate independently. Below is Markov transition graphs for both cases are presented.

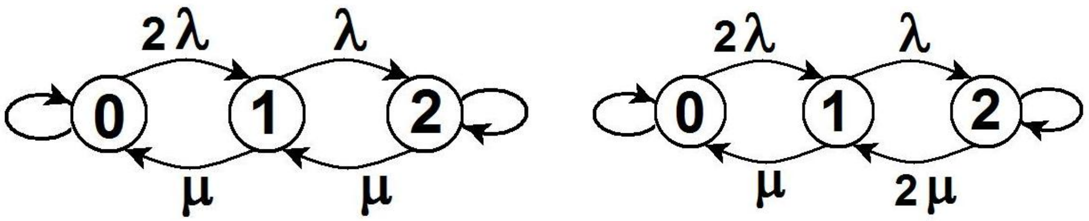


Figure 1.12. Transition graphs for repairable duplicated system with active redundancy for two cases: restricted repair (only one failed unit can be repaired at a time) and unrestricted repair (each failed unit can be repaired independently). The digit in the circle denotes the number of failed units.

With the help of these transition graphs, one can easily write down a system of linear differential equations that can be used for obtaining various reliability indices. Take any two of the three equations:

$$\begin{cases} \dfrac{d}{dt}P_0(t) = -2\lambda P_0(t) + \mu P_1(t) \\ \dfrac{d}{dt}P_1(t) = 2\lambda P_0(t) - (\lambda+\mu)P_1(t) + \mu P_2(t) \\ \dfrac{d}{dt}P_2(t) = \lambda P_1(t) - \mu P_2(t) \text{ for restricted repair} \\ \text{or} \\ \dfrac{d}{dt}P_2(t) = \lambda P_1(t) - 2\mu P_2(t) \text{ for restricted repair} \end{cases} \tag{1.22}$$

and take into account chosen initial conditions.

The availability coefficient for these two cases can be calculated using the following formulas (where $\gamma=\lambda/\mu$):

Table 1.2. Availability coefficient for two repair regimes.

	Formula for availability coefficient, $\tilde{R}$	
	Restricted repair	Unrestricted repair
Strict formula	$\dfrac{1+2\gamma}{(1+\gamma)^2}$	$\dfrac{1+2\gamma}{(1+\gamma)^2+\gamma^2}$
Approximation for γ<<1	$1-\gamma^2$	$1-2\gamma^2$

However, our intent is to present methods of optimal redundancy rather than to give detailed analysis of redundant systems. (Such analysis can be found almost in any book listed in Bibliography to Chapter 1.) Thus we will consider only simplest models of redundant systems, i.e. systems with unrestricted repair.

We avoid strict formulas because they are extremely clumsy; instead we present only approximate ones that mostly are used in practical engineering calculations.

Table 1.3. Approximate formulas for availability coefficient

Type of the redundant group	Approximate formula for availability coefficient, $\tilde{R}$	
	Restricted repair	Unrestricted repair
Group of *n* units	$1-(n!)\cdot\gamma^n$	$1-\gamma^n$

Group of type “*k* out of *n*”	$1-[(n-k+1)!]\cdot\binom{n}{n-k+1}\gamma^{n+1}$	$1-\binom{n}{n-k+1}\gamma^{n+1}$

1.5.Repairable redundant group with standby redundant units

Consider now a repairable group of two units: one active and one standby. Behavior of such redundant group can be described with the help of a renewal process: after a failure of the operating unit a standby unit becomes the newly operating one, while the failed unit after repair becomes a standby one, and so on. System failure occurs when a unit undergoing repair is not ready to replace a now not operating unit that has just failed. The process of functioning of this type of duplicated system is illustrated in Figure 13.

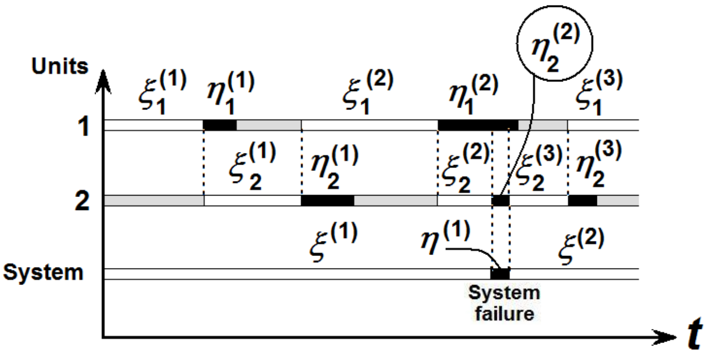


Figure 1.13. Time diagram for a repairable duplicated system with standby redundancy. White parts of a strip denote the operating state of a unit, and grey parts show the standby state, and black parts show the failure state. Here $\xi_j^{(i)}$ denotes *j*-th operating interval of unit *i*, and $\eta_j^{(i)}$ does *j*-th interval of repair of this unit.

In this case, finding PFFO of the duplicated system is also possible with the use of Markov models under assumption of exponentiality of both distributions (of repair time and time to failure),

Transition graphs for restricted and unrestricted repair are shown in the next figure.

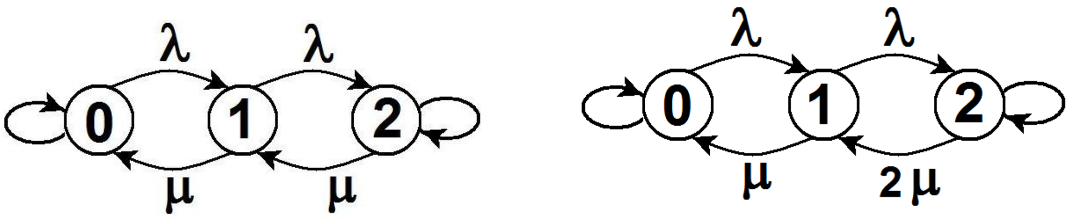


Figure 1.14. Transition graphs for repairable duplicated systems with standby redundancy for two cases: restricted repair (only one failed unit can be repaired at a time) and unrestricted repair (each failed unit can be repaired independently).

Again we present only approximate formulas.

Table 1.4. Approximate formulas for availability coefficient.

Type of the redundant group	Approximate formula for availability coefficient, $\tilde{R}$	
	Restricted repair	Unrestricted repair
Group of *n* units	$1-\gamma^{n-k+1}$	$1-\frac{\gamma^{n-k+1}}{(n-k+1)!}$
Group of type "*k* out of *n*"	$1-(k\gamma)^{n-k+1}$	$1-\frac{(k\gamma)^{n-k+1}}{(n-k+1)!}$

1.6.Multi-level systems and system performance estimation.

Operation of a complex multi-level system cannot be satisfactory described in traditional reliability terms. In this case, one has to talk about performance level of such systems rather than simple binary type "up & down" operating.

Let a system consist of n independent units characterized by their reliability indices $p_1, p_2, .., p_n$. Assume that with unit failure a level of system performance degrades. Denote by Φ_i a quantitative measure of the system performance under condition that unit *i* failed, by Φ_{ij} the same measure if units *i* and *j* are failed, and in general, if some set of units, α is failed then the system performance is characterized by value Φ_α . In this case the system performance can be characterized by the mean value:

$$\Phi_{System} = \sum_{\alpha \in \mathrm{A}} H_\alpha \Phi_\alpha \tag{1.23}$$

where A is a set of all possible states of units 1, 2, …, *n,* i.e. power of this set is 2^n and

$$H_\alpha = \prod_{i \in \alpha} (1-p_i) \prod_{i \in A \backslash \alpha} p_i \ . \tag{1.24}$$

where notation $A \backslash \alpha$ means the total set of unit subscripts with exclusion of subset α.

The measure of system performance could be various: it could be conditional probability of successful fulfillment of operation, productivity, or other operational parameter.

Several years after [Kozlov & Ushakov, 1966] had been published, there was a relative silence with quite rear appearance of works on the topic. Since average measure is not always a good characterization, soon there was a suggestion to evaluate the probability that a multi-state system performance is exceeding some required level. In a sense, it was nothing more than introducing a failure criterion for a multi-state system. In this case, new formulation of the system reliability has the form

$$R_{System} = \Pr\{\Phi_{\alpha} \geq \Phi_{\mathrm{Required}}\} = \sum_{\alpha:\Phi_{\alpha} \geq \Phi_{\mathrm{Required}}} H_{\alpha}\Phi_{\alpha} \ . \tag{1.25}$$

In 1985, in [Ushakov, ed., 1985] Kurt Reinschke introduced a system that itself consists of multi-state units. However, this work also did not find an appropriate response among reliability specialists at the time.

Nevertheless, reliability analysis of multi-state systems has started for all three possible classes:

(1) Multi-state systems consisting of binary units
(2) Binary systems consisting of multi-state units
(3) Multi-state systems consisting of multi-state units

In late 1990s, one observes a real avalanche of papers and since then this topic keeps its steady flow. This subject is considered in more details in Chapter 11.

Naturally, after multi-system analysis, attention to the problems of optimal redundancy in such systems arose. Now the problem of optimal redundancy in multi-state systems is a subject of an intensive research.

1.7. Brief review of other types of redundancy

In reliability theory, redundancy is understood as using additional units for replacement/substitution of failed units. Actually, there are many various types of redundancy. Below we briefly consider structural redundancy, functional redundancy, a system with spare time for operation performance, and so on.

<u>1.7.1.Two-pole structures</u>. One of the typical types of structural redundancy is presented by networks. The simplest network structure is the so-called *bridge structure*. Assume that connection between points *A* and *D* is needed.

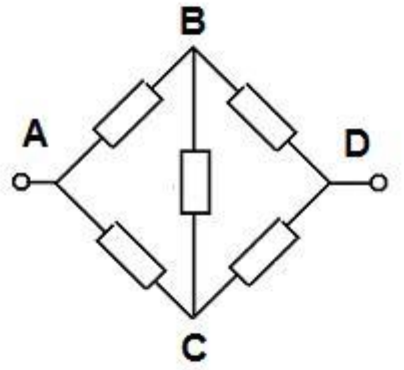


Figure 1.15. Bridge structure.

A failure of any unit does not lead to failure of the system because of the redundant structure. There are the following paths from *A* to *D*: *ABD, ACD, ACBD* and *ABCD*. If at least one of those paths exists, the system performs its task. Of course, one can consider all cuts that lead to the system failure: *AB&AC*, *BD&CD, AB&BC&CD* and *AC&BC&BD*. However, in this case we cannot use simple formulas of series and parallel systems, since paths are interdependent, as well cuts. Because of this, one can only write the upper and lower bounds for PFFO of such systems:

$$\begin{aligned}&(1-Q_{AB}Q_{AC})\cdot(1-Q_{BD}Q_{CD})\cdot(1-Q_{AB}Q_{BC}Q_{CD})\cdot(1-Q_{AC}Q_{BC}Q_{BD})<R_{Bridge}<\\&1-(1-P_{AB}P_{BD})\cdot(1-P_{AC}P_{CD})\cdot(1-P_{AB}P_{BC}P_{CD})\cdot(1-P_{AC}P_{BC}P_{BD}).\end{aligned} \qquad (1.26)$$

For this simple case, one can find a strict solution using a straightforward enumeration of all possible system states:

$$R_{Bridge}=R_{BC}(1-Q_{AB}Q_{AC})\cdot(1-Q_{BD}Q_{CD})+Q_{BC}[1-(1-P_{AB}P_{BD})(1-P_{AC}P_{CD})]. \qquad (1.27)$$

More complex systems of this type are presented by the two-pole networks: in such systems a “signal” has to be delivered from a terminal *A* to terminal *B*. Reliability analysis of such systems usually is performed with the use of Monte Carlo simulation.

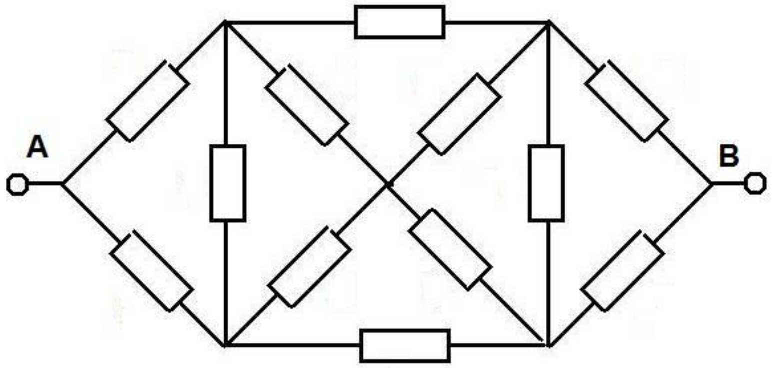


Figure 1.16. An example of two-pole network.

For networks with general structure, the exact value of the reliability index can be found actually only with the help of a direct enumeration. For evaluation of this index one can use the upper and lower bounds of two types: Esary-Proschan boundaries [Barlow & Proscha, 1965] or Litvak-Ushakov boundaries [Ushakov, ed. 1985]. Unfortunately, boundaries cannot be effectively used for solving optimal redundancy problems.

<u>1.7.2. Multi-pole networks</u>. This kind of networks is very common in modern life: one can remember telecommunication networks, transportation and energy grids, etc. The most important specific of such systems is their structural redundancy and redundant capacity of its components. We demonstrate specific of such systems on a simple illustrative example. Consider a bridge structure that was described above, but assume that each node is either a “sender’ or a “receiver” of “flows” to each other. Of course, flows can be different as well as capacities of particular links. Assume that traffic is symmetrical, i.e. traffic from *X* to *Y* is equal to traffic from *Y* to *X*. This assumption allows us to consider only one-way flow between any points.

Let the traffic in the considered network be described as it shown in Table 1.5

Table 1.5. Traffic in the network (in conditional units)

	A	B	C	D
A	-	1	1	1
B	1	-	2	1
C	1	2	-	1
D	1	1	1	-

For normal operating, it is enough to have the following capacities of the links:

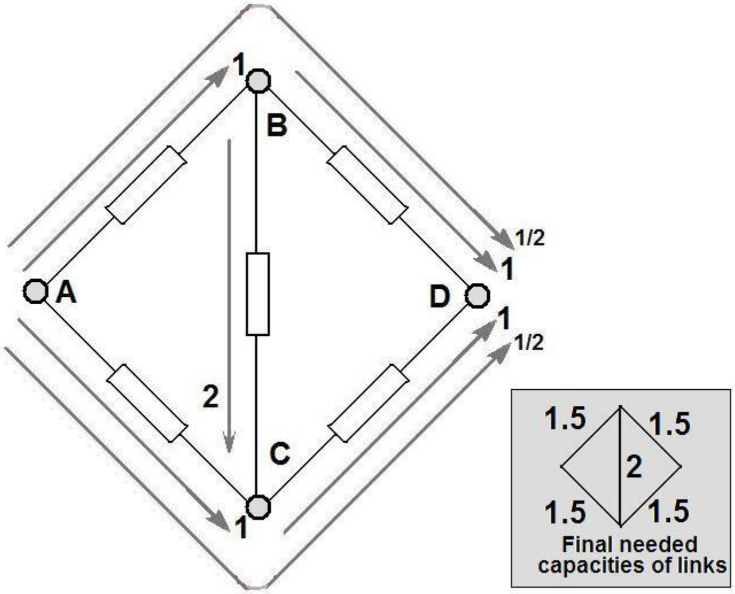


Figure 1.17. Traffic distribution

(We assume that traffic within the network is distributed as uniformly as possible.)

However, links (as well as nodes) are subjected to failure. For protection of the system against link failures, let us consider possible scenarios of link failure and measures of system protection by means of links capacities increase.

What should we do if link *AB* has failed? The flow from *A* to *B* and from *A* to *D* should be redirected. Thus, successful operation of the network requires an increase of the links’ capacities (see the figure below)

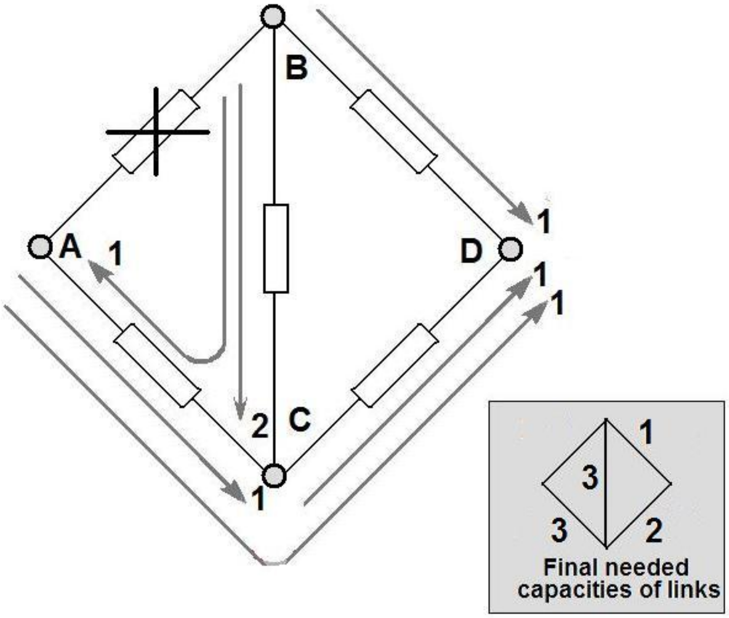


Figure 1.18. Traffic distribution in the case of link *AB* failure.

Since all four outside links are similar, failure of any link *AC*, *BD* or *CD* leads to a similar situation. Thus, to protect the system against failure of any outside link, one should increase capacities of each outside link from 2 to 3 units.

What happens if link *BC* fails? This link originally was used only for connecting nodes *B* and *C*. This traffic should be redistributed: half flow is directed through links *BA*-*AC*, and another – through links *BD*-*DC*. To protect the system against link BC failure, the capacity of each outside link has to be increased by one unit.

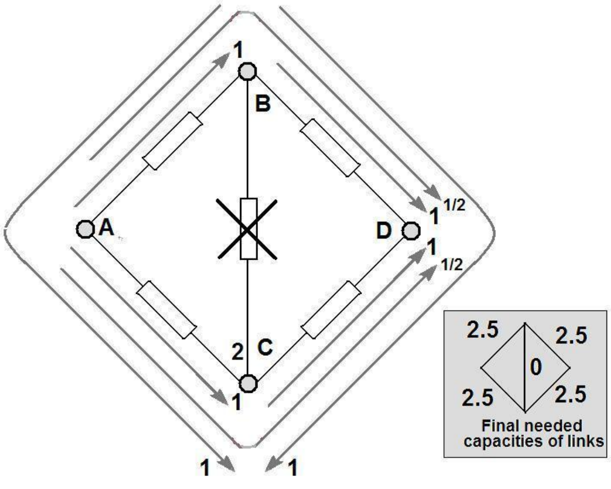


Figure 1.19. Traffic distribution in the case of link *BC* failure.

To protect the system against any single link failure, one has to make link capacities corresponding to maximum at each considered scenarios;

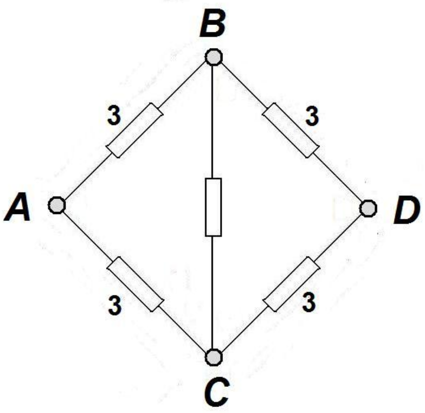


Figure 1.20. Final values of link capacities for a network protected against any possible single failure.

1.7.3. Branching structures. Another rather specific type of redundant systems is presented by systems with a branching structure. In such systems, actual operational units are units on the lowest level, which, however, successfully operate only under the condition that their controlling units at the upper levels are successfully operating. Such structures are very common in various control systems, in particular, in military systems.

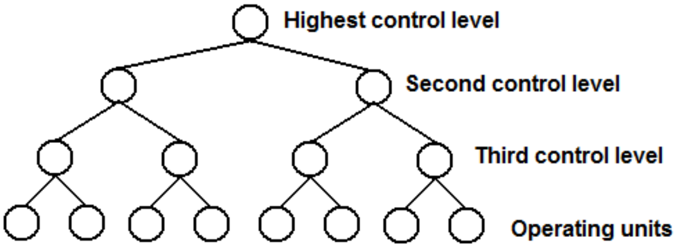


Figure 1.21. System with branching structure.

Assume that the presented branching system performs satisfactory until four or more units of the lower level failed or lost control by upper level units. Types of possible system failures are given in Figure 1.22.

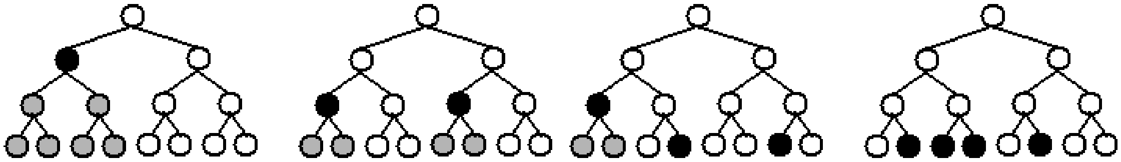


Figure 1.22. Types of situations when the branching system has 4 lower level units that have failed to perform needed operations. (Failed units are in black and units without control are in grey.)

Of course, for complex systems the concept of “failure” is not adequate; instead, there is the notion of diminished performance. For instance, for the same branching system considered above, it is possible to introduce several levels of performance. Assume that the system performance depending on the system state is described by Table 1.6.

Table 1.6. Levels of system performance for various system states.

Qty of failed units of lower level	Conditional level of performance
0	100%
1	99%
2	95%
3	80%
4	60%
5	50%
6	10%
7	2%
8	0%

Usually, for such systems with structural redundancy, one uses the average level of performance. However, it is possible to introduce a new failure criterion and talk about reliability of such system. For instance, under assumption that admissible level of performance is 80% , one came to the situation considered above: the system is considered failed only when four (or more) its lower level units do not operate sufficiently (failed or lost control).

<u>1.7.4.Functional redundancy</u>. Sometimes to increase a probability of successful performance of a system, designers envisage functional redundancy, i.e. make possible to use several different ways of completing a mission. As an example, one can consider procedure of docking a space shuttle with a space station.

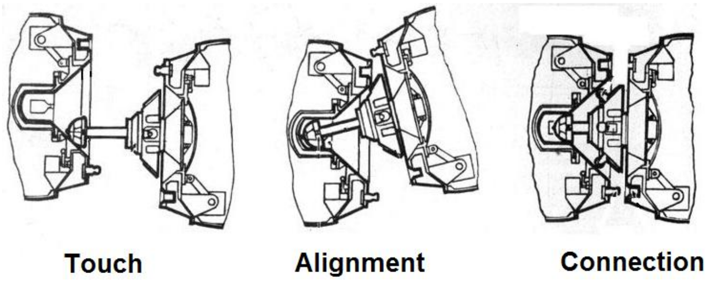


Figure 1.23. Phases of a space shuttle docking to a space station.

This complex procedure can be fulfilled with the use of several various methods: by signals from the ground Mission Control Center (MCC), by on-board computer system and manually. In all these cases, video images sent from space objects are usually used. However, MCC can also use telemetry data. All methods can ensure success of the operation though with different performance.

1.8. Time redundancy

One very specific type of redundancy is the so-called *time redundancy*. There are three main schemes of time redundancy:

(a) A system is operating during interval t_0. There are instantaneous interruptions of the system operation (failures), after which the system starts its operation from the beginning. The system operation is considered successful if during interval t_0 there is at least one interval with length larger than some required value τ. In other words, there is some extra time to restart the operation.

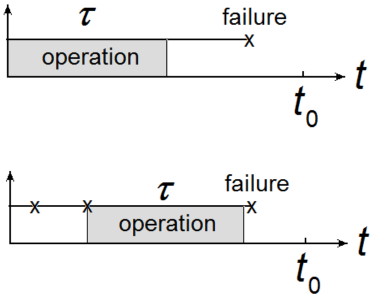


Figure 1.24. Examples of possible implementation of the successful system operation.

Denote the probability of success for such a system by $R(t_0 \mid \tau)$. If there is a failure on interval $[0, t_0]$ at such moment $x< \tau$ that still $t_0 - x > \tau$, the needed operation can be restarted, otherwise $R(t_0 \mid \tau) = 0$. This verbal explanation leads us to the recurrent expression

$$R(t_0 | \tau) \ = R(\tau) + \int_0^{\tau} R(x | t_0 - x) dF(x), \tag{1.28}$$

where $F(x)$ is distribution function of the system time to failure.

Such kind of recurrent equations are usually solved numerically.

(b) Independent of the number of sustained failures, system operation is considered successful if the cumulative time of the system operation is no less than the required amount θ.

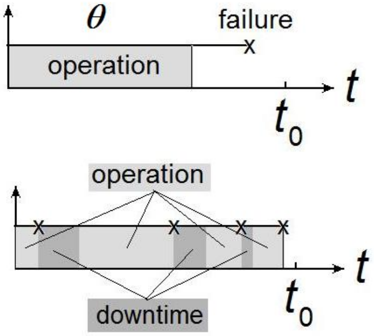


Figure 1.25. Examples of possible implementation of the successful system operation.

Denote the distribution of repair time, η, by $G(t)$. If the first failure has occurred at moment x such that $x > \theta$, it means that the system fulfilled its operation. If failure happens at moment ξ, the system can continue its operation after repair that takes time η, only if $t_0 - \eta >$ θ. It is clear that the probability that the total operating time during interval [0, t_0] is no less than θ is equal to the probability that the total repair time during the same interval is no larger than $t_0 - \theta$.

For this probability, one considers two events that lead to success:

- system works without failures during time θ from the beginning;
- system has failed at the moment $x < t_0 - \theta$, and was repaired during time y, and during the remaining interval of $t_0 - x - y$ accumulates $\theta - x$ units of time of successful operation. This verbal description permits us to write the following recurrent expression:

$$R(t_0 \mid \theta) = 1 - F(t_0) + \int_0^{t_0} \left[\int_0^{t_0 - x} R(t_0 - x - y | \theta) dG(y) \right] dF(x) \tag{1.29}$$

where $R(t_0 \mid z) = 0$ if $z < \theta$.

(c) A system "does not feel" failures of duration less than χ. (In a sense, the system possesses a kind of "inertia" much alike a famous "five second rule".)

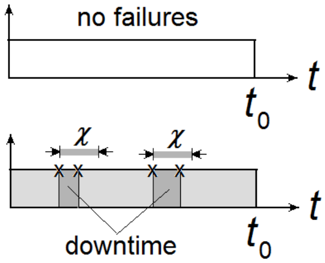


Figure 1.26. Time diagram for a system accumulating operation time.

A system is considered to be successfully operating if during period [0, t_0] there is no down time larger than ψ. This case, in some sense, is a "mirror" of what was considered at the beginning. We will skip explanation details and immediately write the recurrent expression:

$$R(t_0 | \psi) = 1 - F(t_0) + \int_0^{t_0} \left[\int_0^{\eta} R(t_0 - x - y \ | \psi) dG(y) \right] dF(x). \tag{1.30}$$

We will not consider this type of redundancy in details; instead we refer the reader to special literature on the subject [Cherkesov, 1974], [Kredentser, 1978] .

1.9.Some additional optimization problems

1.9.1. Dynamic Redundancy

"*Dynamic redundancy*" models occupy an intermediate place between optimal redundancy and inventory control models.

The essence of a dynamic redundancy problem is contained in the following. Consider a system with *n* redundant units. Some redundant units are operating and represent an active redundancy. These units can be instantly switched into a working position without delay and, consequently, do not interrupt the normal operation of the system. These units have the same reliability parameters (for example, for exponential distribution, and the same failure rate). The remaining units are on standby and cannot fail while waiting. But at the same time, these units can be switched in an active redundant regime only at some pre-determined moments of time. The total number of such switching moments is usually restricted because of different technical and/or economical reasons.

A system failure occurs when at some moment there are no active redundant units to replace the main ones which have failed. At the same time, there may be many standby units which cannot be used because they cannot be instantly switched after a system failure.

Such situations in practice can arise in different space vehicles which are participating in long journeys through the Solar System. A similar situation occurs when one considers using uncontrolled remote technical objects whose monitoring and service can be performed only rarely.

It is clear that if all redundant units are switched to an active working position at an initial moment *t=0*, the expenditure of these units is highest. Indeed, many units might fail in vain during the initial period. At the same time, the probability of the unit's failure during this interval will be small. On the other hand, if there are few active redundant units operating in the interval between two neighboring switching points, the probability of the system's failure decreases. In other words, from a general viewpoint, there should exist an optimal rule (program) of switching standby units into an active regime and allocating these units over all these periods.

Before we begin to formulate the mathematical problem, we discuss some important features of this problem in general.

Goal Function

Two main reliability indices are usually analyzed: the probability of failure-free system operation during some specified interval of time, and the mean time to system failure.

System Structure

Usually, for this type of the problem, a parallel system is under an analytical consideration. Even a simple series system requires a very complex analysis.

Using Active Redundant Units

One possibility is that actively redundant units might be used only during one period after being switched into the system. Afterwards, they are not used further, even if they have not failed. In other words, all units are divided in advance into several independent groups, and each group is working during its own specified period of time. After this period has ended, another group is switched into the active regime. In some sense, this regime is similar to the preventive maintenance regime.

Another possibility is to keep operationally redundant units in use for next stages of operation. This is more effective but may entail some technical difficulties.

Controlled Parameters

As we mentioned above, there are two main parameters under our control: the moments of switching (i.e., the periods of work) and the number of units switched at each switching moment. Three particular problems arise: we need to choose the switching moments if the numbers of switched units are fixed in each stage; we need to choose the numbers of units switched in each stage if the switching moments are specified in advance; and, in general, we need to choose both the switching moments and the numbers of units switched at each stage.

Classes of Control

Consider two main classes of switching control. The first one is the so-called *prior rule (program switching)* where all decisions are made in advance at time *t=0*. The second class is the *dynamic rule* where a decision about switching is made on the basis of current information about a system's state (number of forthcoming stages, number of standby units, number of operationally active units at the moment, etc.).

We note that analytical solutions are possible only for exponentially distributed TTF's. The only possible method of analysis for an arbitrary distribution is via a Monte Carlo simulation.

* * *

Chronological Bibliography of Main Monographs on Reliability Theory (with topics on Optimization).

1962. Lloyd, D.K., and Lipov, M. Reliaility Management, Methods and Mathematics. Prentice Hall.

1965. Barlow, R.E., and F. Proschan. Mathematical Theory of Reliability. John Wiley & Sons.

1966. Kozlov, B.A., and Ushakov, I.A. Brief Handbook on Reliability of Electronic Devices. Sovetskoe radio, 1966.

1967. Raikin, A.L. Elements of Reliability Theory for Engineering Design. Sovetskoe Radio, Moscow,

1968. Polovko, A. M. Fundamentals of Reliability Theory. Academic Press.

1969. Gnedenko, B.V., Belyaev, Yu.K., and Solovyev, A.D. Mathematical Methods in Reliability Theory. Academic Press.

1969. Ushakov, I.A. Method of Solving Optimal Redundancy Problems under Constraints (in Russian). Sovetskoe Radio, Moscow.

1970. Kozlov, B.A., and Ushakov, I.A. Reliability Handbook. Holt, Rinehart and Winston.

1974. Cherkesov, G.N. Reliability of Technical Systems with Time Redundancy (in Russian). Sovetskoe Radio, Moscow.

1975. Barlow, R.E., and F. Proschan. Statistical Theory of Reliability and Life Testing, Holt, Rinehart & Winston.

1975. Gadasin, V.A., Ushakov, I.A. Reliability of Complex Information and Control Systems (*Russian*). Sovetskoe Radio, Moscow.

1975. Kozlov, B.A., and Ushakov, I.A. Handbook of Reliability Calculations for Electronic and Automatic Equipment (in Russian). Sovetskoe Radio, Moscow.

1978. Kozlow, B.A., and Uschakow, I.A. Handbuch zur Berehnung der Zuverlassigkeit in Elektronik und Automatechnik (in German). Academie-Verlag.

1978. Kredentser, B.P. Forcasting Reliability for Time Redundamcy (in Russian). Naukova Dumka. Kiev.

1978. Raikin, A. L. Reliability Theory of Complex Systems (in Russian). Sovietskoe Radio, Moscow.

1979.   Kozlow, B.A., and Uschakow, I.A. Handbuch zur Berehnung der Zuverlassigkeit in Elektronik und Automatechnik (in German). Springer-Verlag.

1980. Tillman, F.A., C.L. Hwang, and Kuo, W. Optimization of System Reliability, *Marcel Dekker,*

1981. Barlow, R.E., and Proschan, F. Statistical Theory of Reliability and Life Testing, 2nd ed. Silver Spring, MD: To Begin With.

1983. Gnedenko, B.V. (Ed.) Aspects of Mathematical Theory of Reliability (in Russian). Radio i Svyaz.

1983. Ushakov, I.A. Textbook on Reliability Engineering (in Bulgarian). VMEI.

1985. Ushakov, I.A., Ed. Handbook on Reliability (in Russian). Radio i Svyaz, Moscow.

1986. Rudenko, Yu.N., and Ushakov, I.A. Reliability of Power Systems (in Russian). Nauka, Moscow.

1987. Reinschke, K., Ushakov, I. Application of Graph Theory for Reliability Analysis (*German*). Verlag Technik, Berlin.

1988. Reinschke, K., and Ushakov, I. Application of Graph Theory for Reliability Analysis (*Russian*). Radio i Svyaz, Moscow.

1988. Reinschke, K., and Ushakov, I. Application of Graph Theory for Reliability Analysis, with K. Reinschke (*German*). Springer, Munchen-Vien.

1989. Rudenko, Yu.N., and Ushakov, I.A. Reliability of Power Systems, 2nd edition (in Russian). Nauka, Moscow.

1991. Kececioglu, D. Reliability Engineering Handbook. Prentice-Hall.

1992. Volkovich, V.L., Voloshin, A.F., Ushakov, I.A., and Zaslavsky, V.A. Models and Methods of Optimization of Complex Systems Reliability (in Russian). Naukova Dumka, Kiev.

1994. Ushakov, I.A. Handbook of Reliability Engineering. John Wiley & Sons.

1995. Gnedenko, B.V., and Ushakov, I.A. Probabilistic Reliability Engineering. John Wiley & Sons.

1997. Kapur, K.C., and Lamberson, L.R. Reliability in Engineering Design. John Wiley & Sons.

1999. Gnedenko, B.V., Pavlov, I.V., and Ushakov, I.A. Statistical Reliability Engineering. John Wiley & Sons.

2003. Kuo, W., and Zuo, M.J. Optimal Reliability Modeling: Principles and Applications, John Wiley & Sons.

2003. Pham, H. Handbook of Reliability Engineering. Springer.

2006. Kuo, W., Prasad, V. R., Tillman, F.A., and Hwang, C.-L. Optimal Reliability Design: Fundamentals and Applications. Cambridge University Press.

2006. Levitin, G., (Ed.). Computational Intelligence in Reliability Engineering. Evolutionary Techniques in Reliability Analysis and Optimization *(*Series: Studies in Computational Intelligence), Vol. 39, Springer-Verlag.

2007. Ushakov, I. A. Course on Reliability Theory (in Russian). Drofa, Moscow.

2010. Gertsbakh, I., and Shpungin, Y. Models of Network Reliability. CRC Press

2. FORMULATION OF THE OPTIMAL REDUNDANCY PROBLEMS

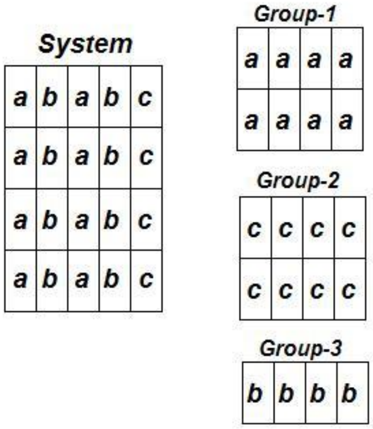
2.1. Problems description

One of the most frequently used methods of reliability increase is the use of additional (redundant) units, circuits and blocks. This method is especially convenient when the principal solution of the system design has already been found: the use of redundant units usually does not cause a change in the overall structure of the system. But the use of extra units entails additional expense. Naturally, a system designer always tries to find the least expensive way to improve reliability. Thus, a designer faces two problems:

Direct problem of optimal redundancy: find such allocation of redundant units among different subsystems that warrants required level of reliability index with spending minimum possible resources;
Inverse problem of optimal redundancy: find such allocation of redundant units among different subsystems that maximizes the level of chosen reliability index under some specified constrains on the total cost of the system.

The choice of a type of constrains depends on the specific engineering problem. Of course, the cost of a set of redundant units is not a unique objective function. For instance, for submarines the most serious constrain is the total volume (or weight) of spare units.

Consider a series system composed of *n* independent redundant groups (or subsystems). A redundant group is not necessarily a separate part of a system. In this context, this may be a group of units of the same type which uses the same redundant units. For instance, in spare parts allocation problems a redundant group might be a set of identical units located throughout the entire system in quite different places.



System

a	b	a	b	c
a	b	a	b	c
a	b	a	b	c
a	b	a	b	c

Group-1

a	a	a	a
a	a	a	a

Group-2

c	c	c	c
c	c	c	c

Group-3

b	b	b	b

Figure 2.1. Modular system and “informal” redundant groups for this system.

2.2. Formulation of the optimal redundancy problem with a single restriction.

The simplest (and practically the most often encountered) optimal redundancy problem is optimization of an objective function under a single constrain. Usually, the following two objective functions are considered: the cost of the total set of redundant units, $C(X)$, and the probability of a failure-free system operation, $R(X)$, where as above X=(x_1, x_2… x_n) and x_i is the number of units within redundant group i.

The *direct problem of optimal redundancy* can be written as:

$$\min_{\mathrm{X}} [C(\mathrm{X}) | R(\mathrm{X}) \geq R_0] \tag{2.1}$$

and the *inverse problem* that can be written as:

$$\max_{\mathrm{X}} [R(\mathrm{X}) | C(\mathrm{X}) \leq C_0] \tag{2.2}$$

where R_0 and C_0 are given constrains for the specific problems.

Cost of redundant group as a whole is usually assumed a linear function of number of redundant units and expressed as

$$C(X) = C(x_1, x_2, ..., x_n) = \sum_{1 \leq i \leq n} C_i \left(x_i \right) \tag{2.3}$$

In most cases, we deal with a system that can be presented as a series connection of independent redundant groups. For such systems, the probability of a successful operation during time t, $R(t|X)$, and availability coefficient, $\tilde{R}(X)$, can be presented as a product of corresponding indices of redundant groups. Because both these objective functions are similar by their probabilistic nature, let us use a common notation $R(X)$ for both cases. Then we can write:

$$R(\mathrm{X}) = R(x_1, x_2, ..., x_n) = \prod_{1 \leq i \leq n} R_i \left(x_i \right) \tag{2.4}$$

where X= (x_1,x_2, ... , x_n) is the set of the system redundant units x_i's of the ith type, $1 \leq i \leq n$, and the $R_i(x_i)$ are the reliability indices of the ith redundant group.

Sometimes it is more convenient to present (3) in an additive form

$$L(\mathrm{X}) = L(x_1, ..., x_n) = \sum_{1 \leq i \leq n} L_i(x_u) \tag{2.5}$$

where $L(X)$ = ln $R(X)$ and $L_i(x_i)$ = ln $L_i(x_i)$.

If a system is highly reliable, i.e.

$$Q_i(x_i) = 1 - R_i(x_i) << \frac{1}{n} \tag{2.6}$$

or, equivalently,

$$Q(X) = Q(x_1,...,x_n) << 1 \ , \tag{2.7}$$

one can use the approximation

$$Q(\mathrm{X}) = Q(x_1, \dots, x_n) \approx \sum_{1 \le i \le n} Q_i(x_i) \tag{2.8}$$

Of course, similar problems can be formulated for other objective functions, for instance, for mean time to failure (or mean time between failures), *T*. Unfortunately, the calculation of *T*(*X*) is usually rather difficult.

Example 2.1

Consider a simplest series system consisting of two units. Unit parameters are given in Table 2.1.

Table 2.1

	PFFO	Cost
Unit-1	0.7	1.2
Unit-2	0.6	2.7

We need to find:

(a) A number of units of both types, $X^{opt} = (x_1^{opt},\ x_2^{opt})$ that satisfy the required level of system reliability index equal to 0.8 and deliver minimum possible system cost.

(b) A number of units of both types, x_1^{opt} and x_2^{opt} that maximize system reliability index under constrain that the total system cost is not higher than 7 units.

It is assumed that "hot" redundancy is used for reliability improvement, i.e. $R_i(x_i) = 1 - q_i^{x_i}$.

Since we don't assume any *a priori* knowledge of optimization methods, let us use a trivial enumerating. For the further convenience, let us introduce *triplets* that contain the following information: $\Delta_i(x_i) = \{x_i, R_i(x_i), C_i(x_i)\}$. Compile two tables with the system cost and PFO, putting in Table 2.2 cost of different variants.

Table 2.2. Values of system cost for various X.

				x_2	
			1	2	3
			2.7	5.4	8.1
	1	1.2	3.9	6.6	9.3
	2	2.4	5.1	7.8	10.5
	3	3.6	6.3	9.0	11.7
	4	4.8	7.5	10.2	12.9
x_1	5	6.0	8.7	11.4	14.1

Table 2.3. Values of system PFFO for various X.

	x_2	
1	2	3

			0.600	0.840	0.936
x_1	1	0.700	0.420	0.588	0.655
	2	0.910	0.546	0.764	0.852
	3	0.973	0.584	0.817	0.911
	4	0.992	0.595	0.833	0.928
	5	0.998	0.599	0.838	0.934

On the basis of these two tables, one can easily compile a new table with triplets ordered by cost.

Table 2.4. List of triplets ordered by the system cost.

$C(X)$	R(X)	$X^{(j)}$	x_1	x_2	
2.7	0.420	(1)	1	1	
3.9	0.546	(2)	2	1	
5.1	0.584	(3)	3	1	
6.3	0.595	(4)	4	1	
6.6	0.588	(5)	1	2	*
7.5	0.599	(6)	5	1	
7.8	0.764	(7)	2	2	
9.0	0.817	(8)	3	2	
9.3	0.655	(9)	1	3	*
10.2	0.833	(10)	4	2	
10.5	0.852	(11)	2	3	
11.4	0.838	(12)	5	2	*
11.7	0.911	(13)	3	3	
12.9	0.928	(14)	4	3	
14.1	0.934	(15)	5	3	

One can see that there are such triplets $X^{(k)}$ and $X^{(k+1)}$ that $C(X^{(k+1)}) > C(X^{(k)})$ but $R(X^{(k+1)}) < R(X^{(k)})$. In this case, it is said that triplet $\Delta^{(k)}$ $(X^{(k)}) = \{ X^{(k)}, R(X^{(k)}), C(X^{(k)})\}$ dominates over triplet $\Delta^{(k+1)}$ $(X^{(k+1)}) = \{ X^{(k+1)}, R(X^{(k+1)}), C(X^{(k+1)})\}$. Such triplets are excluded in further analysis. (In Table 2.4 these vectors are $X^{(5)}$, $X^{(9)}$ $X^{(12)}$.) All remaining vectors X$^{(k)}$ are called *dominating*.

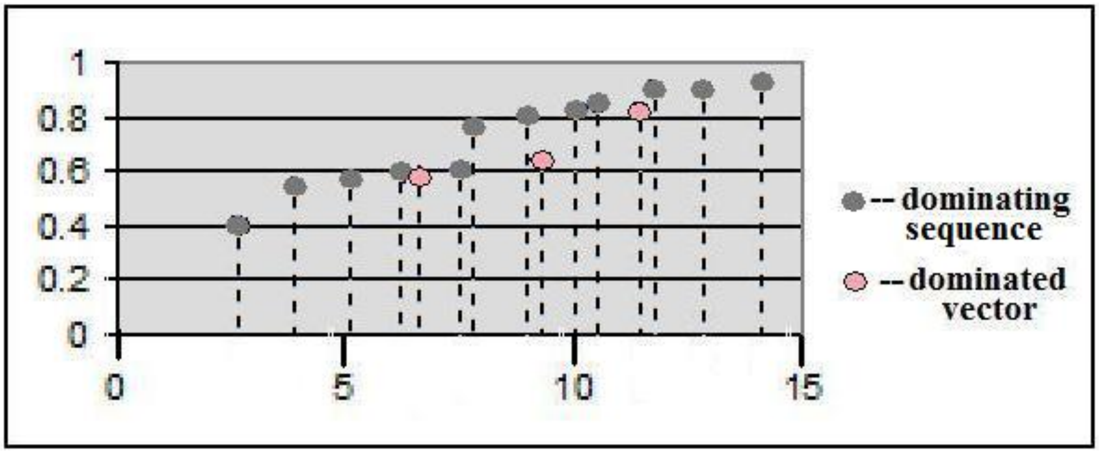


Figure 2.2. Dominating sequence and dominated vectors for the numerical example.

Basing on Table 2.4, one can easily find desired solutions:

(a) for the direct optimal redundancy problem, one finds the largest value of cost that is still admissible;

(b) for the inverse optimal redundancy problem, one finds the smallest value of reliability index that exceeds required value.

For considered numerical example the direct problem solution is vector $X^{(8)}$ (reliability index = 0.817 and system cost = 9) and for the inverse problem solution is vector $X^{(4)}$ (system cost = 6.3 and reliability index = 0.595).

2.3.FORMULATION OF OPTIMAL REDUNDANCY PROBLEM WITH MULTIPLE CONSTRAINS

2.3.1 Direct optimal redundancy problem.

Sometimes the optimal redundancy problem is formulated for multiple constrains, for instance, maximization of a reliability index under condition that other factors (cost, volume, weight, etc.) are limited by some fixed conditions. This problem can be written as:

$$\max_X \left\{ R(X) \middle| C_1(X) \ge C_1^0, C_2(X) \ge C_2^0, \dots, C_M(X) \ge C_m^0 \right\} \qquad (2.9)$$

where C_j^0, $j = 1, 2, \dots, M$, are given constrains on the corresponding type of expenditures for the system as a whole.

In this case, further detailed considerations as in the section above are possible though we omit them for the sake of brevity.

2.3.2. Inverse optimal redundancy problem.

Very rarely one can find the following problem: a system is designated for multiple tasks and performing each task requires different parts of the system. Set of such system parts may be called subsystems. Some parts of the system are used for all tasks and some only for performance specific tasks. Tasks of these subsystems may have different reliability requirements (for instance, some subsystems may perform extraordinary responsible tasks).

For such systems, one can formulate the following problem

$$\min_X \left\{ C(X) \middle| R_1(X) \ge R_1^0, R_2(X) \ge R_2^0, \dots, R_M(X) \ge R_m^0 \right\} \qquad (2.10\ )$$

To make the problem clearer, consider a simple illustrative example where a system of four units is conditionally depicted as three interdependent subsystems.

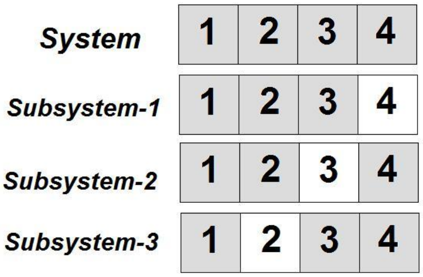


Figure 2.3. Conditional dividing a system by subsystems.

For this system as a whole the inverse problem of optimal redundancy can be written as:

$$\min_{X}\left\{C(x)\,\middle|\, r_1(x_1)r_2(x_2)r_3(x_3)\geq R_1^0,\quad r_1(x_1)r_2(x_2)r_4(x_4)\geq R_2^0,\quad r_1(x_1)r_3(x_3)r_4(x_4)\geq R_3^0\right\}. \tag{2.11 }$$

<u>Example 2.2.</u> Consider the same system as in Example 2.1. Introduce one more unit parameter, say, weight, W. Unit parameters are given in Table 2.5.

Table 2.5.

	PFFO	Cost	Weight
Unit-1	0.7	1.2	2.3
Unit-2	0.6	2.7	1.5

We need to find a number of units of both types, $X^{opt} = (x_1^{opt},\ x_2^{opt})$ that maximize system reliability index under constrains on both limiting factors: C(X) $\leq$ 7 units of cost and W(X) $\leq$ 10 units of weight., that is we consider the inverse optimization problem

For solution, use Tables 2.6 and 2.7 and add to them a new one for total system weight.

Table 2.6. Values of system weight for various X.

				x_2	
			1	2	3
			1.5	3	4.5
x_1	1	2.3	5	7.7	10.4
	2	4.6	7.3	10	12.7
	3	6.9	9.6	12.3	15
	4	9.2	11.9	14.6	17.3
	5	11.5	14.2	16.9	19.6

On the basis of this and previous Table, let us compile a new table (now with quadruples, since we have four parameters) with triplets ordered by increase of cost.

Table 2.7. List of triplets ordered by the system cost.

$C(X)$	W(X)	$R(X)$	$X^{(j)}$	x_1	x_2	

2.7	5.0	0.420	(1)	1	1	
3.9	7.3	0.546	(2)	2	1	
5.1	9.6	0.584	(3)	3	1	
6.3	11.9	0.595	(4)	4	1	
6.6	7.7	0.588	(5)	1	2	optimum
7.5	14.2	0.599	(6)	5	1	
7.8	10.0	0.764	(7)	2	2	
9.0	12.3	0.817	(8)	3	2	
9.3	10.4	0.655	(9)	1	3	*
10.2	14.6	0.833	(10)	4	2	
10.5	12.7	0.852	(11)	2	3	
11.4	16.9	0.838	(12)	5	2	*
11.7	12.3	0.911	(13)	3	3	
12.9	17.3	0.928	(14)	4	3	
14.1	16.9	0.934	(15)	5	3	

In this table all cells with inadmissible cost or weight are shadowed. Thus, the maximum reachable level of reliability index under the given constrains is 0.588; and it is reached by vector $X^{(5)}$. It is interesting to notice that in the previous example this vector was dominated and could not be a solution.

In case of multi-constrain situation, dominating sequence also exists. For instance, vector $X^{(12)}$ is dominated by vector $X^{(11)}$: for larger values $C(X)$=11.4 and $W(X)$ = 16.9 the reliability index, $R(X)$ = 0.838 that is smaller than 0.852. Another such pair of vectors is $X^{(9)}$ and $X^{(7)}$: both parameters "cost-weight" for $X^{(9)}$ (9.3; 10.4) are correspondingly larger than analogous parameters for $X^{(7)}$, though the latter vector is characterized by larger reliability index. (All dominated vectors are marked with "*".)

We don't supply a numerical example for direct problem solution due to its clumsiness.

2.4. Formulation of multi-criteria optimal redundancy problems.

2.4.1. Direct multi-criteria optimal redundancy problem .

Assume that a designer has to reach the required level of reliability having several limiting factors like cost, weight, volume, etc. Usually all these factors are somehow dependent: a miniature units can be more expensive, weight and volume of a unit are naturally dependent, etc. What does it mean to say "the best solution" in this case? Solutions satisfying the same reliability requirements can be incomparable: one variant have smaller total weight, etc.

Actually, the problem of choosing a preferable solution lies outside the scope of mathematics: it is up to a decision maker. However, there are some useful procedures for finding the so-called *non-improvable solutions*. It means that none of the selected variants (solutions) is strictly better than another but it is chosen in accordance to some subjective measures of preference.

A set of the multi-criteria problem solutions is called the *Pareto set*. In mathematical terms one can write the problem in the form:

$$\underset{X}{\mathrm{MIN}}\left\{C_1(X), C_2(X), \ldots, C_M(X) \mid R(X) \geq R^0\right\} \tag{2.12}$$

where the symbol MIN in capital letters denotes Pareto "minimization" of vector $\{C_1(X), C_2(X), \ldots, C_M(X)\}$.

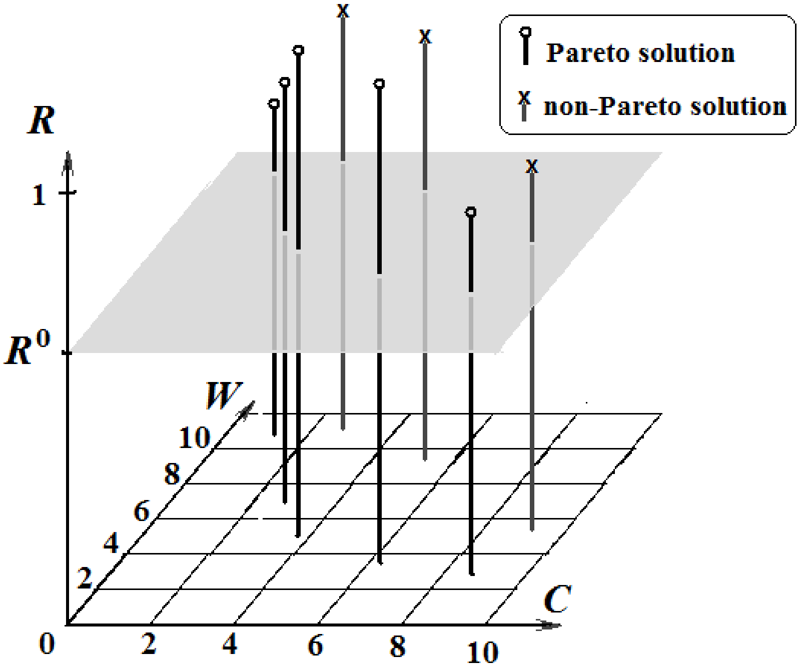


Figure 2.4. Explanation of the Pareto solutions for direct multi-criteria problem.

All Pareto solutions for condition $R(X) \geq R^0$ are dominating in a vector sense: for each Pareto-optimal vector $X^{(k)}$, there is no vector $X' < X^{(k)}$ that R(X') ≥ R($X^{(k)}$).

2.4.2. Inverse multi-criteria optimal redundancy problem.

The inverse problem for multi-criteria case can be written as:

$$\underset{X}{\mathrm{MAX}}\left\{R_1(X), R_2(X), \ldots, R_M(X) \mid C(X) \leq C^0\right\} \tag{2.13}$$

where the symbol MAX in capital letters denotes Pareto "maximization" of vector $\{R_1(X), R_2(X), \ldots, R_M(X)\}$.

For instance, for system depicted in Figure 2.3 one can write:

$$\underset{X}{\mathrm{MAX}}\left\{R_1(X),\ R_2(X),\ R_3(X) \mid C(X) \leq C^0\right\} \tag{2.14}$$

Decision about what variant of the system configuration should be chosen has to be done by a decision maker. Expert opinion cannot be formalized: this is why experts still survive in modern computer age! ☺

Bibliography to Chapter 2

(books are distinguished by bold fonts)

1956. Moskowitz, F., and Mclean, J. B. Some reliability aspects of system design. IRE Transactions on Reliability and Quality Control, No.3.

1959. Black, G., and Proschan,F. On Optimal Redundancy. Operations Research, No. 5.

1959. Ghare, P. M., and Taylor, R. E. Optimal redundancy for reliability in series systems. Operations Research, No. 5.

1960. Proschan, F. On optimal supply. Naval Research Logistics Quarterly, No.4.

1961. Morrison, D.F. The Optimum Allocation of Spare Components in Systems.

1963. Blitz, M. Optimum allocation of spare budget. Naval Research Logistics Quarterly, No.2.

1966. Gertsbakh, I.B. Optimum use of reserve elements. *Soviet Journal of Computer and System Sciences*, No. 5.

1966. Lawler, E. L., and Bell, M. D. A method for solving discrete optimization problems. Operations Research, No. 14.

1967. Alekseev, O.G. On one problem of optimal redundancy. Engineering Cybernetics, No. 1.

1967. Alekseev, O.G. Optimization of Complex Systems Reliability under Several Constrains. Automatics and Telemechanics, No.12.
1967. Raikin, A.L., and Mandel, A.S. Construction of the Optimal Schedule of redundant units switching. Automation and Remote Control, No.5.

1968. El-Neweihi, E., Proschan, F., and Sethuraman, J. Optimal allocation of components in parallel-series and series-parallel systems. Journal on Applied Probability, No. 3.

1968. Mizukami, K. Optimum Redundancy for Maximum System Reliability by the Method of Convex and Integer Programming. Operations Research, No. 2.

1969. Ushakov, I.A. Method of Solving Optimal Redundancy Problems under Constraints (in Russian). Sovetskoe Radio, Moscow.

1970. Jensen, P.A. Optimization of Series-Parallel-Series Networks. Operations Research, No. 3.

1970. Messinger, M., and Shooman, M.L. Technique for optimal spare allocation. IEEE Transactions on Reliability, No. 11.

1974. Shaw, L., and Sinkar , S.G. Redundant spares allocation to reduce reliability costs.

Naval Research Logistics Quarterly, No.2.

1975. Agarwal, K. K., Gupta, J. S. and Misra, K. B. A new heuristic criterion for solving a redundancy optimization problem. IEEE Transactions on Reliability,No. 24.

1976. Aggarval, K. V. Redundancy optimization in general systems. IEEE Transaction on Reliability, No. 5.

1977. Nakagawa, Y., and Nakashima, K. A heuristic method for determining reliability allocation. IEEE Transactions on Reliability, No. 26.

1977. Tillman,F.A., Hwang, C.L., and Kuo , W. Determining Component Reliability and Redundancy for Optimum System Reliability. IEEE Transactions on Reliability, No. 26.

1978. Kuo, W., Hwang, C.L., and Tillman, F.A. A note on heuristic methods in optimal system reliability". IEEE Transactions on Reliability, No.5.

1981. Nakagawa, Y., and Miyazaki, S. An experimental comparison of the heuristic methods for solving reliability optimization problems, IEEE Transaction on Reliability, No. 30.

1985. Bulfin, R.L., and Liu , C.Y. Optimal allocation of redundant components for large systems. IEEE Transactions on Reliability, No. 3.

1985. Malashenko, Yu.E., Shura-Bura, A.E., and Ushakov, I.A. Optimal Redundancy. In Handbook: Reliability of Technical Systems (Ed. I. Ushakov), Sovetskoe Radio, Moscow.

1987. Dinghua, S. A new heuristic algorithm for constrained redundancy-optimization in complex systems. IEEE Transaction on Reliability, No. 36.

1991. Boland, P.J., El-Neweihi, E., and Proschan, F. Redundancy importance and allocation of spares in coherent systems. Journal of Statistical Planning and Inference, No. 1-2.

1992. Yanagi, S., Aso, K., and Sasaki, M. Optimal spare allocation problem based on the interval estimate of availability. International Journal of Production Economics, No. 3.

1993. El-Neweihi, E., and Sethuraman, J. Optimal Allocation under Partial Ordering of Lifetimes of Components. Advances in Applied Probability, No. 4

1993. Kim, J. H., and Yum, B. J. A heuristic method for solving redundancy optimization problems in complex systems. IEEE Transaction on Reliability, No. 42.

1996. Jianping, L. A bound heuristic algorithm for solving reliability redundancy optimization., Microelectronics and Reliability, No.3.

1997. Levitin, G., Lisnianski, A., and Elmakis, D. (1997) Structure Optimization of Power System with Different Redundant Elements. Electric Power Systems Research, No. 43.

1997. Rubinstein, R.Y., Levitin, G., Lisnianski, A., and Ben Haim, H. Redundancy optimization of static series-parallel reliability models under uncertainty. IEEE Transactions on Reliability, No. 4.

1998. Prasad V.R., and Raghavachari, M. Optimal allocation of interchangeable components in a series-parallel system. IEEE Transactions on Reliability, No. 1.

1999. Lisnianski, A., Levitin, G., Ben Haim, H., and Elmakis, D. Power system optimization subject to reliability constraints. Electric Power Systems Research, No.39.

1999. Mi, J. Optimal active redundancy allocation in k-out-of-n system. Journal on Applied Probability, No.3.

2003. Kuo, W., and Zuo, M.J. Optimal Reliability Modeling: Principles and Applications, John Wiley & Sons.

2006. Ha, C., and Kuo, W. Reliability redundancy allocation: An improved realization for nonconvex nonlinear programming problems. European Journal of Operational Research, No.1.

2006. Kim, H-G., Bae, C.-O., and Park, D.-J. Reliability-redundancy optimization using simulated annealing algorithms. Journal of Quality in Maintenance Engineering, No. 4.

2006. Kuo, W., Prasad, V. R., Tillman, F.A., and Hwang, C.-L. Optimal Reliability Design: Fundamentals and Applications. Cambridge University Press.

2007. Dai Y., and Levitin, G. Optimal Resource Allocation for Maximizing Performance and Reliability in Tree-structured Grid Services. IEEE Transactions on Reliability, No. 3.

2007. Kuo, W., and Wan, R. Recent Advances in Optimal Reliability Allocation. Computational Intelligence in Reliability Engineering, No. 1.

2007. Liang, Y.-C., Lo, M.-H., and Chen, Y.-C. Variable neighborhood search for redundancy allocation problems. IMA Journal of Management Mathematics , No.2.

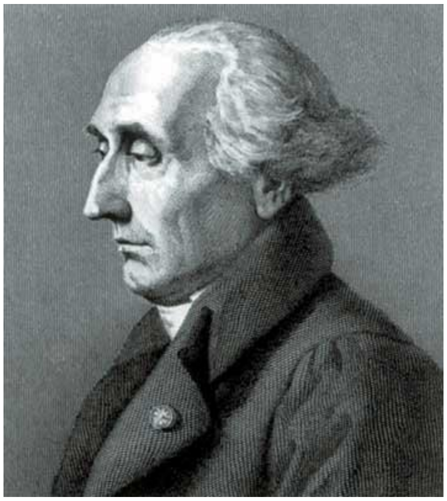
2007. Liang,Y.-C., and Chen, Y.-C. Redundancy allocation of series-parallel systems using a variable neighborhood search algorithm. Reliability Engineering & System Safety, No. 3.

2007. Tavakkoli-Moghaddam, R. A New Mathematical Model for a Redundancy Allocation Problem with Mixing Components Redundant and Choice of Redundancy Strategies. Applied Mathematical Sciences, No. 45.

3.METHOD OF LAGRANGE MULTIPLIERS

One of the first attempts to solve the optimal redundancy problem was based on the classical *Lagrange Multipliers Method*. This method has been invented and developed by great French mathematician Lagrange.

Joseph-Louis Lagrange
(1736 – 1813)



Lagrange made outstanding contributions to all fields of analysis, to number theory, and to classical and celestial mechanics.

Lagrange was one of the creators of the calculus of variations. He introduced the method of Lagrange multipliers where possible constraints were taken into account. Lagrange invented the method of solving differential equations known as variation of parameters and applied differential calculus to the theory of probabilities.

He studied the three-body problem for the Earth, Sun, and Moon and the movement of Jupiter's satellites.

Above all he impressed on mechanics, having transformed Newtonian mechanics into a branch of analysis, Lagrangian mechanics.

This method allows to get extremum value of the function under some specified constrain on another involved function in the form of equality. The Lagrange Multiplier method is applicable if both functions (optimizing and constraining) are monotone and differentiable.

Strictly speaking, this method is not appropriate for optimal redundancy problems solving because the system reliability and cost are described by functions of discrete arguments x_i (numbers of redundant units), and the restrictions on accessible resources (or on required values of reliability) are fixed in the form of inequalities.

Nevertheless, this method is interesting in general and also gives us some useful hints for appropriate solution of some practical problems of discrete nature.

Let us begin with the direct optimal redundancy problem. For solving this problem, we construct the Lagrange Function, $\mathcal{L}(X)$:

$$L(\mathrm{X}) = C(\mathrm{X}) + \Lambda R(\mathrm{X}), \tag{3.1}$$

where $C(X)$ and $R(X)$ are the cost of the system redundant units and the system reliability, respectively, if there are X redundant units of all types, $X = (x_1, x_2, \ldots, x_n)$.

The goal is to minimize $C(X)$ taking into account constrain in the form $R(X^{\mathrm{opt}})=R^0$. Thus, the system of equations to be solved is

$$\begin{cases} \dfrac{\partial \mathcal{L}(\mathrm{X})}{\partial x_i} = \dfrac{\partial C(\mathrm{X})}{\partial x_i} + \Lambda \dfrac{\partial R(\mathrm{X})}{\partial x_i} = 0 \\ \text{for all } i = 1, 2, \dots, n, \text{ and} \\ R(\mathrm{X}^{opt}) = R^0 \end{cases} \tag{3.2}$$

The values to be found are: the x_i^{opt}, i=1, 2,...,n, and Λ.

If both function $C(X)$ and $R(X)$ are separable[2] and differentiable, the first n equations of (3.2) can be rewritten as

$$\begin{cases} \dfrac{d\mathcal{L}_1(x_1)}{dx_1} = \dfrac{dC_1(x_1)}{dx_1} + \Lambda \dfrac{dR_1(x_1)}{dx_1} = 0 \\ \dfrac{d\mathcal{L}_n(x_n)}{dx_n} = \dfrac{dC_n(x_n)}{dx_n} + \Lambda \dfrac{dR_n(x_n)}{dx_n} = 0 \\ R(\mathrm{X}^{opt}) = R^0 \end{cases} \tag{3.3}$$

On a physical level, (3.3) means that for separable functions $L(X)$ and $C(X)$, the optimal solution corresponds to equality of relative increments of reliability of each redundant group for an equal and infinitesimally small resources investment.

$$\frac{dC_1(x_1)}{dx_1} \cdot \left(\frac{dR_1(x_1)}{dx_1} \right)^{-1} = \frac{dC_2(x_2)}{dx_2} \cdot \left(\frac{dR_2(x_2)}{dx_2} \right)^{-1} = \dots = \frac{dC_n(x_n)}{dx_n} \cdot \left(\frac{dR_n(x_n)}{dx_n} \right)^{-1} = -\Lambda. \tag{3.4}$$

In the general case, (3.3) yields no closed form solution but it is possible to suggest an algorithm for numerical calculation:

1) At some arbitrary point $x_1^{(1)}$ calculate the derivative for some fixed redundant group, say, the first one:

$$\left. \frac{dR_1(x_1)}{c_1 dx_1} \right|_{x_1^{(1)}} = -\Lambda^{(1)}. \tag{3.5}$$

<u>**Remark**</u>. Of course, one would like to choose a value of $x_1^{(1)}$ close to an expected optimal solution x_i^{opt}. For example, if you consider spare parts for equipment to operate failure-free during time t and know that the unit MTTF is T, this value should be a little larger than t/T. In other words, this choice should be done based on engineering experience and intuition.

2) For the remaining redundant groups, calculate derivatives until the following condition is satisfied:

$$\left. \frac{dR_i(x_i)}{c_i dx_i} \right|_{x_i^{(1)}} = -\Lambda^{(1)} \ . \tag{3.6}$$

[2] Taking logarithm of multiplicative function $R(X)$, one get an additive function of logarithms of multipliers.

3) Calculate value

$$R(X^{(1)}) = \sum_{1\le i\le n} R_i(x_i^{(1)}) \ . \qquad (3.7)$$

4) Compare $R^{(1)}$ with R^0. If $R^{(1)} > R^0$ choose $x_i^{(2)}<x_i^{(1)}$; if $R^{(1)} < R^0$ choose $x_i^{(2)}>x_i^{(1)}$. After choosing a new value of $x_i^{(2)}$, return to step 1 of the algorithm.

The stopping rule: $X^{(N)}$ is accepted as the solution if the following condition holds

$$\left|R(X^{(N)}) - R^0\right| \le \varepsilon \qquad (3.8)$$

where ε is some specified admissible discrepancy in the final value of the objective function $R(X)$.

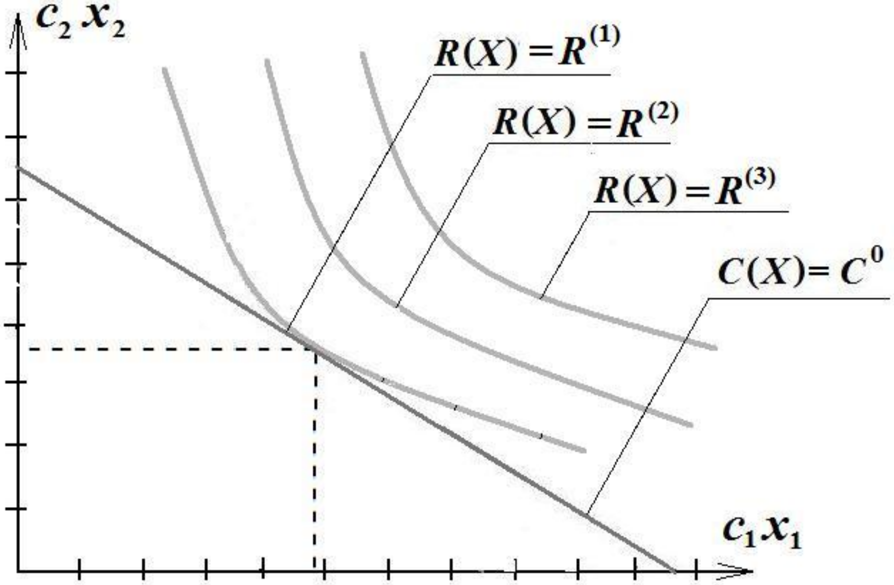


Figure 3.1. Procedure of solving the Direct Problem of Optimal Redundancy.

Solution of the inverse optimal redundancy problem is analogous. Lagrange function is

$$L*(\mathrm{X}) = R(\mathrm{X}) + \Lambda * C(\mathrm{X}) \qquad (3.9)$$

and the equations are:

$$\begin{cases} \dfrac{\partial \pounds(\mathrm{X})}{\partial x_i} = \dfrac{\partial R(\mathrm{X})}{\partial x_i} + \Lambda \dfrac{\partial C(\mathrm{X})}{\partial x_i} = 0 \\ \text{for all } i = 1, 2, \dots, n, \text{ and} \\ C(\mathrm{X}^{opt}) = C^0 \end{cases} \qquad (3.10)$$

The optimal solution has to be got in respect to goal function $C(X)$. The stopping rule in this case:

$$\left|C(X^{(N)}) - C^0\right| \le \varepsilon * \qquad (3.11)$$

where $\varepsilon *$ is some specified admissible discrepancy in the final value of the goal function

C(*X*).

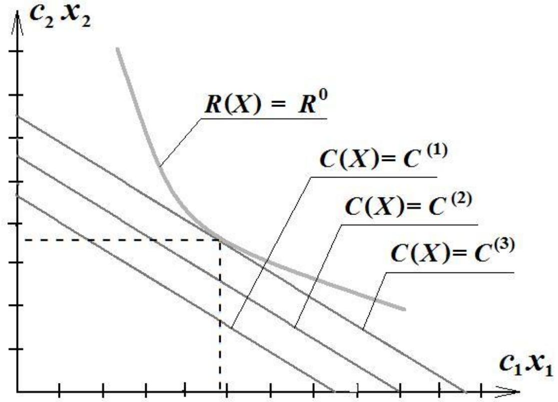


Figure 3.2. Procedure of solving the Inverse Problem of Optimal Redundancy.

Unfortunately, there is only one case where the direct optimal redundancy problem can be solved in a closed form. This is the case of a highly reliable system with active redundancy where

$$R(X) = \prod_{1 \le i \le n} \left(1 - q_i^{x_i}\right) \approx 1 - \sum_{1 \le i \le n} q_i^{x_i} \tag{3.12}$$

or

$$Q(X) \approx \sum_{1 \le i \le n} q_i^{x_i}\ , \tag{3.13}$$

i.e. instead of maximizing *R*(*X*), one can minimize *Q*(*X*) and gets the needed optimal solution.

Taking into account that both objective functions are separable, (3.10) can be written as

$$\begin{cases} \\ \dfrac{dq_i^{x_i}}{dx_i} + \Lambda c_i = 0 \\ \text{for all } i = 1, 2, \ldots, n, \text{and} \\ Q\,(\mathrm{X}^{opt}) = Q^0 \end{cases} \tag{3.14}$$

From equation

$$\frac{dq_i^{x_i}}{dx_i} = \ln q_i \cdot q_i^{x_i}\ , \tag{3.15}$$

we can finally write

$$q_j^{x_j} = \frac{c_j}{\Lambda(-\ln q_j)} \tag{3.16}$$

Now returning to the last equation in (3.10), we have

$$Q^0 = \frac{1}{\Lambda}\sum\frac{c_j}{(-\ln q_j)} \tag{3.17}$$

and, consequently, the Lagrange multiplier has the form

$$\Lambda = \frac{1}{Q^0}\sum\frac{c_j}{(-\ln q_j)} \tag{3.18}$$

From (3.16), one has

$$x_j = \frac{1}{\ln q_i}\cdot\ln\left[\frac{c_j}{\Lambda(-\ln q_j)}\right] \tag{3.19}$$

Finally, after substitution of (3.17) into (3.19), one gets

$$x_j = \frac{1}{\ln q_i}\cdot\ln\left[\frac{c_j}{\Lambda(-\ln q_j)}\right]\cdot\left(\frac{1}{Q^0}\sum\frac{c_j}{(-\ln q_j)}\right)^{-1} \tag{3.20}$$

Hardly anybody would be happy to deal with such a clumsy formula!

The solutions obtained by the Lagrange Multiplier Method are usually continue due to requirements to the objective functions. Immediate questions arise: Is it possible to use an integer extrapolation for each non-integer x_i? If this extrapolation is possible, is the obtained solution optimal?

Unfortunately, even if one tries to “correct” non-integer solutions by substitution lower and upper integer limits $\underline{x_j} < x_j < \overline{x_j}$, this very rarely leads to optimal solution! Moreover, enumerating all 2^n possible “corrections” can itself be a problem. We demonstrate this statement on a simple example.

* * *

Bibliography to Chapter 3

1963. Everett, H. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. Operations Research, No. 3.

1970. Greenberg, H.J. An Application of a Lagrangian Penalty Function to Obtain Optimal Redundancy . Technometrics, No. 3 .

1978. Kuo, W., Lin, H.H., Xu, Z., and Zhang, W. (1987) Reliability optimization with the Lagrange-multiplier and branch-and-bound technique. IEEE Transactions on Reliability, No.5.

1979. Hwang, C. L., Tillman, F. A., and Kuo, W. Reliability Optimization by

Generalized Lagrangian-Function and Reduced-Gradient Methods, IEEE Transactions on Reliability, No.28.

4.STEEPEST DESCENT METHOD

4.1.The main idea of SDM

The *Steepest Descent Method (SDM)* is based on the very natural idea: moving from an arbitrary point in the direction of the maximal gradient of the goal function, it is possible to reach the maximum of a multi-dimensional unimodal function. The origin of the method's name lies in the fact that a water drop runs down on non-flat surface choosing the direction of instantaneous maximum descending.

Probably the next simple example explains the algorithm more graphically. Suppose that a traveler comes to a hill that is hidden in a thick mist. His target is to reach the hill top with no knowledge about the mountain shape except the fact that the hill is smooth enough (has no ravines or local hills). The traveler sees only a very restricted area around the starting point. The question is: What is the shortest path from the initial point to the mountain's top? Intuition hints that the traveler has to move in the direction of the maximal possible ascent at each point on his path to the mountain's top. This direction coincides with the gradient of the function at each point.

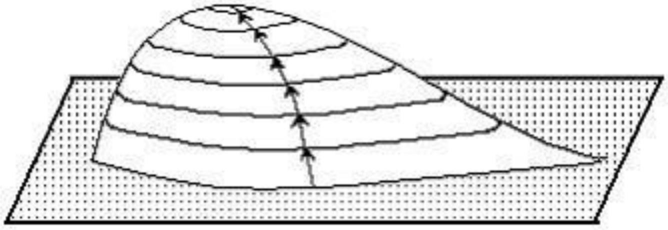


Figure 4.1. A path of a traveler up to the hill top.

However, optimal redundancy problem has an integer nature: redundant units can be added to the system one by one. The previous analogy is useful in case of continuous functions of continuous arguments. But in the case of optimal redundancy, all arguments are discrete. If continue the analogy with a traveler, one sees that there are restrictions on the traveler's movement: he can move only in the North-South or East-West directions and can change the direction only at the vertices of a discrete grid with specified steps. This means that at each vertex one should use the direction of the largest partial derivative. Because of this, one sometimes speaks of the *Method of Coordinate Steepest Descent.*

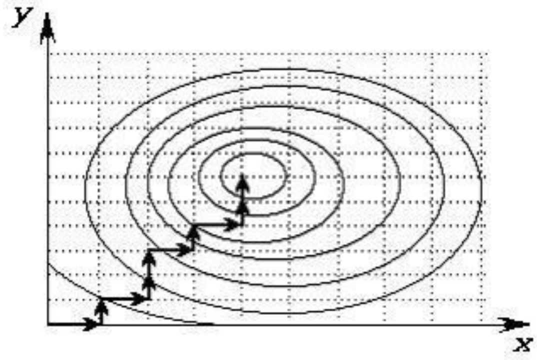


Figure 4.2. A Traveler path when only N-S and E-W directions are allowed.

This idea of finding the maximum of a unimodal function may be applied to the optimal redundancy problem.

4.2.Description of the algorithm

Consider a system that is a series connection of independent redundant groups. Let us use the both goal functions in the additive form

$$L(\mathrm{X}) = \ln R(\mathrm{X}) = \sum_{1\le i\le n} \ln R_i(x_i) \qquad (4.1)$$

and

$$C(\mathrm{X}) = \sum_{1\le i\le n} c_i\, x_i. \qquad (4.2)$$

It is clear that maximization of function *R*(*X*) corresponds to minimization of function *L*(*X*), i.e. optimum solution X^{opt} for goal function *L*(*X*) delivers as well optimum for function *R*(*X*).

Introduce vector $X^{(N)} = (x_1^{(N)}, ..., x_n^{(N)})$, where $x_i^{(N)}$ is the number of redundant units of the *i*th redundant group at the *N*th step of the SDM process. Denote by $R_i(x_i^{(N)})$ reliability index and by $C_i(x_i^{(N)})$ the cost of the *i*th redundant group after the *N*th step of the SDM process. For convenience of further exposition let us introduce also the following additional notation: $X_i^{(N)} = (x_1^{(N)},..., x_{i-1}^{(N)}, 0,\ x_{i+1}^{(N)},..., x_n^{(N)}$, i.e. vector $X_i^{(N)}$ is vector *X* without component x_i. Obviously, $X^{(N)} = X_i^{(N)} + x_i^{(N)}$.

At the *Nth* step of the process, one adds a redundant unit of such type k, for which relative increase of reliability index is maximum, i.e.

$$\gamma_k^{(N)}(x_i^{(N)}) = \max_{1\le i\le n} \frac{\ln R(x_i^{(N)}) - \ln R(x_i^{(N)}+1)}{c_i} \qquad (4.3)$$

A unit of this type is added to the set of system's redundant units. The process continues in the same manner until the optimal solution is obtained.

Now let us describe the optimization algorithm step by step from the very beginning.

1) Before the beginning of the process, there is no redundant units at all, i.e.

$x_1^{(0)} = ... = x_n^{(0)}$ or, in other words, $X^{(0)} = \vec{0}$.

2) For all *i*, *i*=1, 2, …, *n*, one calculates values $\gamma_i^{(0)}(x_i^{(0)})$;
3) One finds such index *k* that delivers maximum

$$\gamma_k^{(0)}(x_k^{(0)}) = \max_{1 \le i \le n} \gamma_i^{(0)}(x_i^{(0)})$$

4) One calculates a new value

$$x_k^{(1)} = x_k^{(0)} + 1$$

5) All other $x_i^{(0)}$, $i \ne k$, change their superscripts: $x_i^{(0)} \Rightarrow x_i^{(1)}$;
6) One gets a new vector of the system's redundant units:

$$X^{(1)} = (X_k^{(0)} + x_k^{(1)})$$

7) One calculates the value of $L(X^{(1)})$ and determines the corresponding $R(X^{(1)})$;
8) One calculates the value of $L(X^{(1)})$;
9) One calculates by the same rule a new value $\gamma_k^{(1)}(x_k^{(1)})$;
10) All other values $\gamma_i^{(0)}(x_i^{(0)})$, $i \ne k$, are conserved but one changes their superscripts:

$$\gamma_i^{(0)}(x_i^{(0)}) \Rightarrow \gamma_i^{(1)}(x_i^{(1)});$$

11) GOTO (3).

4.3.The stopping rule

The solution of the direct problem of optimal redundancy is reached at such step N, for which the following condition is valid

$$C(X^{(N)}) \le C^0 < C(X^{(N+1)}). \qquad (4.4)$$

The value of $R(X^{(N)})$ is the maximum possible for the given constrain on the system cost.

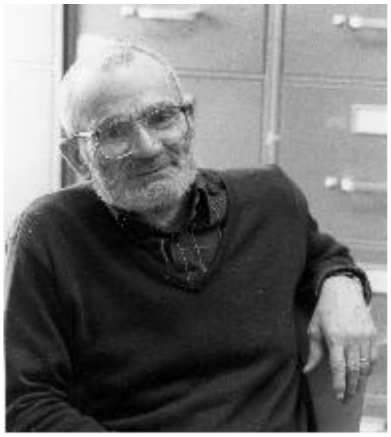
Remark. Sometimes the SDM procedure requires to add at the last step a very expensive unit but adding this unit exceed the given constrain on the total cost of the system's redundant units. At the same time, if one does not add this unit, there are some extra financial resources to add other, less expensive units. In this case one may bypass the expensive unit and continue the procedure.

The inverse optimal redundancy problem reaches its optimal solution at such step *N* that the following condition is valid:

$$R(X^{(N-1)}) < R^0 \le R(X^{(N)}). \qquad (4.5)$$

The value of $C(X^{(N)})$ is the minimum possible for the given constrain on the required system reliability index.

American mathematician Frank Proschan [Barlow&Proschan, 1965, 1981] has proven that the SDM procedure delivers members of dominating sequence if each function $R_i(x_i)$ is concave.



Frank Proschan
(1921-1993)

Frank Proschan is an American mathematician. He earned his Ph.D. in statistics from Stanford University in 1959. He had held positions with the Federal Government at the National Bureau of Standards (1941-1952), with Sylvania Electric Products (1952-1960) and with Boeing Scientific Labs (1960-1970). Since 1970 he had been Professor of Statistics at Florida State University.

He had many honors including the Von Neumann Prize award presented by TIMS-ORSA. He was a Fellow of Institute of the American Statistical Association and a member of International Statistical Institute.

Example 4.1. A series system consists of six different units whose parameters are given in Table 4.1. (Distributions of time to failure are assumed exponential.)

Table 4.1. Units' parameters.

Unit type	Number of units, n_k	Unit failure rate, λ_k (10^{-5} 1/hr)	Unit cost,c_k
1	5	1	1
2	10	1	1
3	5	1	8
4	10	1	8
5	5	8	1
6	10	8	1

One needs to find the optimum number of standby units for successful system operation during t_0=1000 hrs for two cases:

(1) Required PFFO is 0.9995;
(2) Admissible expenses on all spare units are 40 cost units.

First of all, using Table 4.1, let us find parameters of Poisson distributions for each group by the formula $a_i = \lambda_i n_i t_0$:

Table 4.2. Values of parameters of Poisson distribution

Parameter	Value
a_1	$1 \cdot 10^{-5} \cdot 5 \cdot 1000 = 0.05$
a_2	$1 \cdot 10^{-5} \cdot 10 \cdot 1000 = 0.1$
a_3	$1 \cdot 10^{-5} \cdot 5 \cdot 1000 = 0.05$
a_4	$1 \cdot 10^{-5} \cdot 10 \cdot 1000 = 0.1$

a_5	$8\cdot 10^{-5}\cdot 5\cdot 1000=0.4$
a_6	$1\cdot 10^{-5}\cdot 10\cdot 1000=0.8$

The probability of appearance of exactly *k* failures during given time t_0 is calculated by the formula:

$$q_i(k)=\frac{(\lambda_i n_i t_0)^k}{k!}\exp(-\lambda_i n_i t_0). \tag{4.6}$$

By the way, if condition $Q_i(x_i)>>Q_i(x_i+1)$ holds, then calculation of values γ can be simplified up to

$$\gamma_i(x_i)\approx\frac{q_i(x_i)}{c_i}. \tag{4.7}$$

Calculated values of $q_i(x_i)$ are presented in Table 4.3.

Table 4.3. Values of unreliability indices for various x_i.

x_i	$q_1(x_1)$	$q_2(x_2)$	$q_3(x_3)$	$q_4(x_4)$	$q_5(x_5)$	$q_6(x_6)$
0	0.0476	0.0905	0.0476	0.0905	0.268	0.359
1	0.00119	0.00452	0.00119	0.00452	0.0536	0.144
2	1.98E-05	0.000151	1.98E-05	0.000151	0.00715	0.0383
3	2.48E-07	3.77E-06	2.48E-07	3.77E-06	0.000715	0.00767
4	2.48E-09	7.54E-08	2.48E-09	7.54E-08	5.72E-05	0.00123
5	2.06E-11	1.26E-09	2.06E-11	1.26E-09	3.81E-06	0.000164
6	1.47E-13	1.8E-11	1.47E-13	1.8E-11	2.18E-07	1.87E-05
7	9.22E-16	2.24E-13	9.22E-16	2.24E-13	1.09E-08	1.87E-06
8	5.12E-18	2.49E-15	5.12E-18	2.49E-15	4.84E-10	1.66E-07
9	2.56E-20	2.49E-17	2.56E-20	2.49E-17	1.94E-11	1.33E-08

Now we can build the next table where values $\gamma_i(x_i)$ are presented.

Table 4.4. Values $\gamma_i(x_i)$ for all redundant groups.

x_i	$\gamma_1(x_1)$	$\gamma_2(x_2)$	$\gamma_3(x_3)$	$\gamma_4(x_4)$	$\gamma_5(x_5)$	$\gamma_6(x_6)$
1	**2** 0.0464	*1* 0.086	**8** 0.0058	*6* 0.0107	*4* 0.0268	*3* 0.027
2	**11** 0.00117	*9* 0.00437	*16* 0.000146	*14* 0.000547	*7* 0.00581	*5* 0.0132
3	*19* 1.96E-05	*15* 0.000147	*24* 2.45E-06	*20* 1.84E-05	*13* 0.000804	*10* 0.003834
4	*28* 2.45E-07	*23* 3.69E-06	3.07E-08	*26* 4.62E-07	*18* 8.22E-05	*12* 0.000805

5	2.46E-09	7.41E-08	3.07E-10	9.27E-09	*22* 6.67E-06	*17* 0.000133
6	2.05E-11	1.24E-09	2.56E-12	1.55E-10	*27* 4.49E-07	*21* 1.81E-05
7	1.47E-13	1.77E-11	1.83E-14	2.22E-12	2.59E-08	*25* 2.10E-06
8	9.16E-16	2.22E-13	1.15E-16	2.77E-14	1.30E-09	*29* 2.13E-07
9	5.09E-18	2.47E-15	6.37E-19	3.09E-16	5.81E-11	1.91E-08
	. . .	. . .	. . .	. . .	. . .	. . .

In this table, the numbers in the upper right corner of cells are numbers corresponding to steps of the SDM procedure.

On the basis of Table 4.4, let us build the final table, from which one can get needed optimal solutions.

Table 4.5. Step-by-step results of SDM procedure

	C(X)	Q(X)	x1	x2	x3	x4	x5	x6
…	…	…	…	…	…	…	…	…
10	24	0.021871	1	2	1	1	2	3
11	25	2.07E-02	2	2	1	1	2	3
12	26	1.43E-02	2	2	1	1	2	4
13	27	7.83E-03	2	2	1	1	3	4
14	35	3.46E-03	2	2	1	2	3	4
15	36	3.31E-03	2	3	1	2	3	4
16	44	2.14E-03	2	3	2	2	3	4
17	45	1.07E-03	2	3	2	2	3	5
18	46	4.16E-04	2	3	2	2	4	5
19	47	3.96E-04	3	3	2	2	4	5
20	55	2.49E-04	3	3	2	3	4	5
21	56	1.03E-04	3	3	2	3	4	6
22	57	5.01E-05	3	3	2	3	5	6
23	58	4.64E-05	3	4	2	3	5	6
24	66	2.69E-05	3	4	3	3	5	6
25	67	1.00E-05	3	4	3	3	5	7
26	75	6.33E-06	3	4	3	4	5	7
…	…	…	…	…	…	…	…	…

The last table allows finding optimal solutions for both optimal problems: direct as well as inverse.

By conditions of the illustrative problem, R^0=0.9995. From Table 4.5, we find that the solution is reached at Step 18: unreliability in this case is 4.16E-04, i.e. $R(X^{\text{opt}})$ = 0.999584 that satisfies the requirements. (Corresponding cost is 46 cost units.)

At the same time for the inverse problem the solution is reached at Step 15 with the total cost of 36 units. (Corresponding PFFO=0.99669.) In this case we keep extra 4 cost units. Of course, they could be spent for 4 additional inexpensive units of types 1, 2, 5 and 6, i.e. instead of obtained solution $(x_1 = 2,\ x_1 = 3,\ x_1 = 1,\ x_1 = 2,\ x_1 = 3,\ x_1 = 4)$ take solution with all resources spent: $(x_1 = 3,\ x_1 = 4,\ x_1 = 1,\ x_1 = 2,\ x_1 = 4,\ x_1 = 5)$. In this case, the system PFFO is equal to 0.99844. This solution is admissible in sense of the total cost of redundant units.

By the way, to get such a solution we could slightly change the SDM algorithm: If on a current step of the SDM procedure we “jump” over the admissible cost, we can take another unit or units with admissible cost.

Analogous corrections could be performed if the obtained current solution for the direct problem of optimal redundancy overexceeds the required value R^0.

4.5.Approximate solution

For practical purposes, an engineer sometimes needs to know an approximate solution which would be close to an optimal one *a priori*. Such a solution can be used as the starting point for the SDM calculation procedure. (Moreover, sometimes the approximate solution is a good pragmatic solution if input statistical data are too unconfident. Indeed, attempts to use strong methods with unreliable input data might be considered as a total absence of common sense! Remember: the “garbage-in-garbage-out” rule is valid for precise mathematical models as well!).

The proposed approximate solution [Ushakov, 1965] is satisfactory for highly reliable systems. This means that in the direct optimal redundancy problem the value of Q^0 is very small. In a sense, such a condition is not a serious practical restriction. Indeed, if the investigated system is too unreliable, one should question if it is reasonable to improve its reliability at all. Maybe it is easier to find another solution, for instance, to use another system?

For a highly reliable system one can write

$$Q^0 \approx Q(\mathrm{X}^{(N)}) \approx \sum_{1\le i\le n} Q_i\left(x_i^{(N)}\right) \tag{4.8}$$

at the stopping moment (the *N*th step) of the optimization process when the value of the reliability index should be high enough.

From Table 4.4, one can see that there is some “strip” that divided all values of $\gamma_i(x_i)$. For instance, consider cells corresponding to steps 19-24 (shadowed on the table). The largest value laying above this “strip” (step 18) has value γ = 8.22E-05 that is larger than any value of γ belonging to the “strip”. At the same time, the largest value laying below the “strip” (step 25) has value $\gamma = 2\cdot10^{-6}$ that is smaller than any corresponding value on the “strip”. It means that there is some value Λ, $2\cdot10^{-6} < \Lambda < 8.22\cdot10^{-5}$ that divides all set of γ in two specific subsets: this Λ in a sense play the role of Lagrange multiplier. Indeed, approximate equality of γ for each "argument" x_i completely corresponds to the equilibrium in the Lagrange solution.

Let us make a reasonable assumption that at the stopping moment

$$\gamma_1^{(N)} \approx \gamma_2^{(N)} \approx \ldots \approx \gamma_n^{(N)} \approx \Lambda \tag{4.9}$$

At the same time,

$$\gamma_i^{(N)} \approx \frac{Q_i(x_i^{(N)})}{c_i} \approx \Lambda \tag{4.10}$$

Now using (4.8)

$$Q^0 \approx \Lambda \sum_{1 \le i \le n} c_i \tag{4.11}$$

and finally

$$\Lambda \approx \frac{Q^0}{\sum_{1 \le i \le n} c_i} \tag{4.12}$$

Now we can substitute (4.12) into (4.10) and obtain:

$$Q_i(x_i^{(N)}) \approx \frac{c_i Q^0}{\sum_{1 \le i \le n} c_i} \tag{4.13}$$

For solving the inverse optimal redundancy problem, one has to use a very simple iterative procedure.

(1) Find approximate starting values of the x_i's

$$x_1^{(1)} = x_2^{(1)} = \ldots = x_n^{(1)} = \frac{C^0}{\sum_{1 \le i \le n} c_i} \tag{4.14}$$

(2) Use these x_i's to calculate $Q^{(1)}$ as

$$Q^{(1)} = \sum_{1 \le i \le n} Q_i\left(x_i^{(1)}\right) \tag{4.15}$$

(3) Calculate $g^{(1)}$ as

$$\gamma^{(1)} = \frac{Q^{(1)}}{\sum_{1 \le i \le n} c_i} \tag{4.16}$$

(4) For $\gamma^{(1)}$ determine $x_i^{(2)}$ for all i from the equations

$$Q_i(x_\mathrm{i}^{(2)}) = c_i\, \gamma^{(1)}\ . \tag{4.17}$$

(4) For all obtained $x_\mathrm{i}^{(2)}$, one calculates the total cost of the system's redundant units as:

$$C^{(2)} = \sum_{1\le i\le n} c_i\, x_i^{(2)}. \tag{4.18}$$

5) If $C^{(1)} > C^0$ one sets a new $\gamma^{(2)} > \gamma^{(1)}$, if $C^{(1)} < C^0$ one sets a new $\gamma^{(2)} < \gamma^{(1)}$.

After this, the procedure continues from the 3rd step. Such iterative procedure continues until the appropriate value of the total cost of redundant units is achieved.

* * *

Bibliography to Chapter 4

(books are distinguished by bold font)

1965. Barlow, R.E., and F. Proschan. Mathematical Theory of Reliability. John Wiley & Sons.

1965. Ushakov, I. A. Approximate solution of optimal redundancy problem. Radiotechnika, No. 12.

1967. Ushakov, I.A. On optimal redundancy problems with a non-multiplicative objective function. Automation and Remote Control, No. 3.

1969. Ushakov, I.A. Method of Solving Optimal Redundancy Problems under Constraints (in Russian). Sovetskoe Radio, Moscow.

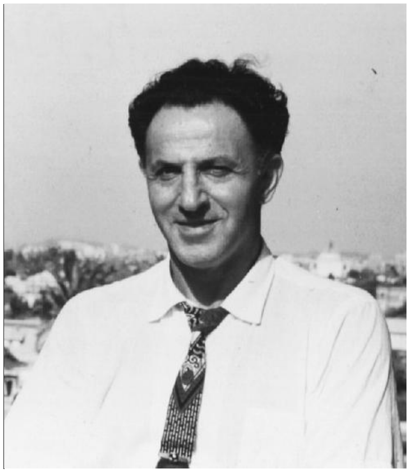
1981. Ushakov, I.A. Methods of Approximate Solution of Dynamic Standby Problems. Engineering Cybernetics (USA), Vol.19, No.2.

5.DYNAMIC PROGRAMMING

As mentioned above, the problem has an essentially discrete nature, so the SDM cannot guarantee the accuracy of the solution. Thus, if an exact solution of the optimal redundancy problem is needed, one generally needs to use the *Dynamic Programming Method* (*DPM*).

5.1. Bellman's Algorithm

Main ideas of the DPM were formulated by an American mathematician Richard Bellman [Bellman, 1957], who has formulated the so-called optimality principle**.**



Richard Ernest Bellman
(1920 – 1984)

American applied mathematician, who is famous for his invention of dynamic programming in 1953. He also made many important contributions in other fields of mathematics.

Over the course of his career he published 619 papers and 39 books. During the last 11 years of his life he published over 100 papers despite suffering from crippling complications of a brain surgery.

Bellman's fundamental contributions to science and engineering had won him many honors: First Norbert Wiener Prize in Applied Mathematics (1970).

The DPM provides an exact solution of discrete optimization problems. In fact, it is a well organized method of direct enumeration. For the accuracy of the solutions one has to pay with a high calculation time and a huge computer memory if the problem is highly dimensional.

To solve the direct optimal redundancy problem, let us construct a sequence of Bellman's function, $B_k(r)$. This function reflects the optimal value of the goal function for a system of k redundant groups and a specified restriction r. As usual, start in the beginning ☺:

$$B_1(r^{(1)}) = \min_{x_1} \left\{ c_1 x_1 \,\middle|\, R_1(x_1) \ge r^{(1)};\ 0 < r^{(1)} \le R^0 \right\}. \tag{5.1}$$

It is clear that in such a way we determine the number of units that is necessary for the redundant group to have reliability index equal to $r^{(1)}$ that is laying within interval [0, R^0].

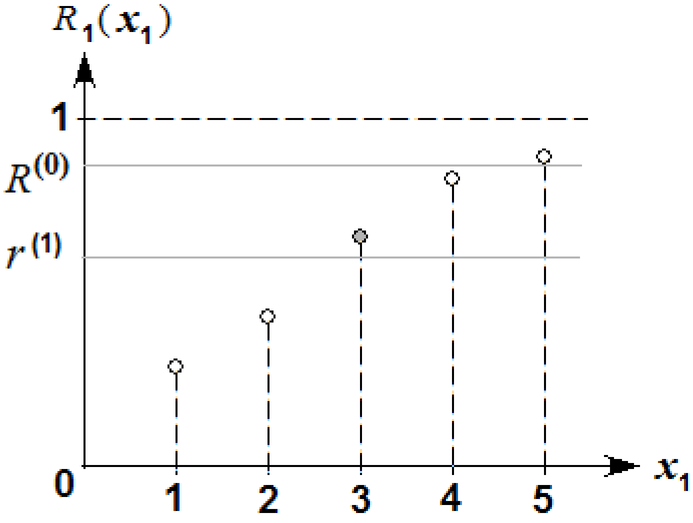


Figure 5.1. Illustration of the solution for Equation (5.1).

Now compose the next function

$$B_2(r^{(2)}) = \min_{x_2} \left\{ c_2 x_2 + B_1(r^{(1)}) \;\middle|\; r^{(1)} \cdot R_2(x_2) \geq R^0 \right\}. \tag{5.2}$$

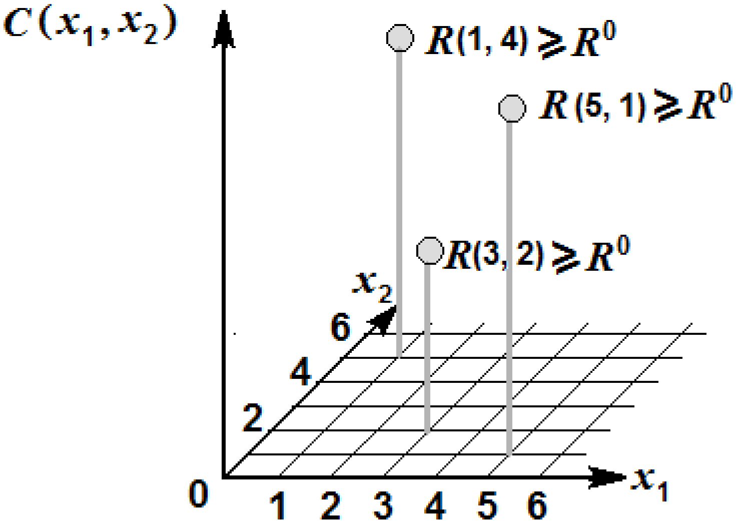


Figure 5.2. Illustration of the solution for Equation (5.2).

In a sense, we have a “convolution” of the first and second redundant groups and for each level of current redundancy the best variant of such convolution is kept for the next stage of the procedure. In analogous way the recurrent procedure continues until the last Bellman’s equation is compiled:

$$B_n(r^{(n)}) = \min_{x_n} \left\{ c_n x_n + B_{n-1}(r^{(n-1)}) \,\middle|\, r^{(n-1)} \cdot R_n(x_n) \ge R^0 \right\}. \tag{5.3}$$

Actually, Equation (5.3) gives us only a solution for x_n : other x_i's are "hidden in previous stages of compiling Bellman's equation. Indeed, Equation (5.3) contains $B_{n-1}(r^{(n-1)})$, which allows us to determine x_{n-1}, and so on. The last found will be x_1. In a sense, the process of finding optimal x_i*'s* is going backwards relating to the process of Bellman's function compiling.

Solution of the inverse problem of optimal redundancy is similar. The only difference is that the system reliability becomes an objective function and the total system cost becomes the constraint.. The procedure does not need additional explanations.

At the first stage of the recurrent procedure, one compiles the Bellman's equation of the form:

$$\tilde{B}_1(c^{(1)}) = \max_{x_1} \left\{ R_1(x_1) \,\middle|\, c_1 x_1 \le c^{(1)};\ 0 < c^{(1)} \le C^0 \right\}. \tag{5.4}$$

Then consequently other equations:

$$\tilde{B}_2(c^{(2)}) = \max_{x_2} \left\{ R_2(x_2) \cdot \tilde{B}_1(c^{(1)}) \,\middle|\, c_2 x_2 + c^{(1)} \le C^0 \right\}. \tag{5.5}$$

$$\cdots$$

$$\tilde{B}_n(c^{(n)}) = \max_{x_n} \left\{ R_n(x_n) \cdot \tilde{B}_{n-1}(c^{n-1)}) \,\middle|\, c_n x_n + c^{(n-1)} \le C^0 \right\}. \tag{6.6}$$

Optimal solution is found by the same backward returning to the beginning of the recurrent procedure.

For illustration of the calculated procedure of dynamic programming, let us consider a simple illustrative example.

Example 5.1.

Let us consider a very simple series system consisting of three units with the characteristics: $p_1 = 0.7$, $p_2 = 0.8$, $p_3 = 0.9$, and $c_1 = c_2 = c_3 = 1$. For reliability increase, a "hot" redundancy is used.

The problem is to find the optimal vector of redundant units for the system under constrain: $C(X) \le 6$ cost units.

The table with all possible convolutions of redundant groups 1 and 2 is presented below. Dominating vectors within each are marked with symbols *y*.

Table 5.1. Convolution of redundant groups 1 and 2.

Cost	x_1	x_2	$R(x_1, x_2)$	Chosen
0	0	0	0.56	y_0
1	1	0	0.728	y_1
	0	1	0.672	
2	2	0	0.7784	
	1	1	0.8736	y_2

	0	2	0.6944	
3	3	0	0.79352	
	2	1	0.93408	y_3
	1	2	0.90272	
	0	3	0.69888	
4	4	0	0.798056	
	3	1	0.952224	
	2	2	0.965216	y_4
	1	3	0.908544	
	0	4	0.699776	
5	5	0	0.799417	
	4	1	0.957667	
	3	2	0.983965	*y5*
	2	3	0.971443	
	1	4	0.909709	
	0	5	0.699955	
6	6	0	0.799825	
	5	1	0.9593	
	4	2	0.989589	
	3	3	0.990313	y_6
	2	4	0.972689	
	1	5	0.909942	
	0	6	0.699991	
. . .	. . .	. . .	. . .	**. . .**

Now compile a table with only dominating sequence for convolution of redundant groups 1 and 2, denoting each pair as y_1, y_2, etc.

Table 5.2.

y_k	$R(y_k)$	x_3	$R_3(x_3)$	R_{syst}	Chosen
y_0	0.56	6	≈ 1	0.56	
y_1	0.728	5	0.999999	0.727999	
y_2	0.8736	4	0.99999	0.873591	
y_3	0.93408	3	0.9999	0.933987	
y_4	0.965216	2	0.999	0.964251	
y_5	0.983965	1	0.99	0.974125	X
y_6	0.990313	0	0.9	0.891282	

Thus, the solution is (x_3, y_5). Now return to Table 5.1 and find there that y_5 corresponds to $x_1 = 3$ and $x_2 = 2$. This is the final step of the solving procedure.

The solution of the direct problem of optimal redundancy is more complicated for manual calculations, however it can be easily programmed for a computer.

5.2.Kettelle's Algorithm

Actually, DPM is a well organized enumeration using convolutions of a set of possible solutions. It has some “psychological” deficiency: a researcher gets the final results without “submerging” into the solving process. If a researcher is not satisfied by a particular solution for some specified restrictions and decides to change them, it may lead to a complete re-solving of the problem.

John D. Kettelle, Jr.
(1925-2012)

John Kettelle was an American mathematician who fought for 3 years in WWII in US Navy and then served 2 years on a submarine during Korean war.

Next 5 years he had been working in Operations Research group at Arthur D. Little Co. with the founder of Operations Research George Kimbell. Later he started a series of consulting corporations.

He was the author of a well known paper on modified Dynamic Programming method. He edited 11 books published by ORSA.

Recently he has developed a method of negotiations with computer as the third party.

For most practical engineering problems, using the *Kettelle's Algorithm* [Kettelle, 1962] is actually a modification of the DPM. It differs from DPM by a simple and intuitively clear organization of calculating process. This algorithm is very effective for the exact solution of engineering problems due to its clarity and flexibility of calculations.

Of course, the *Kettelle's Algorithm,* as well as DPM, requires more computer time and memory than the SDM, but it gives strict solutions. At the same time, this algorithm allows to construct entire dominating sequence (as SDM), that gives a possibility to switch from solving the direct optimal redundancy problem to the inverse one using the previously calculated sequence.

5.2.1. General description of the method.

We shall describe the Kettelle's Algorithm step by step.

(1) For each *i*th redundant group, one constructs a table of values of $R_i(x_i)$, accompanied by corresponding cost $C_i(x_i)$.

Table 5.3. Initial dominating sequences for redundant groups

Group	Number of redundant units in the group					
number	0	1	2	…	n	…
1	$R_1(0)$, $C_1(0)$,	$R_1(1)$, $C_1(1)$,	$R_1(2)$, $C_1(2)$,	…	$R_1(0)$, $C_1(0)$,	…
2	$R_2(0)$, $\mathrm{C}_2(0)$	$R_2(1)$, $\mathrm{C}_2(1)$	$R_2(2)$, $\mathrm{C}_2(2)$	…	$R_2(0)$, $\mathrm{C}_2(0)$	…
…	…	…	…	…	…	…
N	$R_N(0)$, $C_N(0)$	$R_N(1)$, $C_N(1)$	$R_N(2)$, $C_N(2)$	…	$R_N(0)$, $C_N(0)$	…

The sequence of these pairs for each group forms a *dominating sequence*, i.e. for any j and k: $R(k) < R(k+1)$ and $C(k) < C(k+1)$.

(2) Take any two redundant groups from Table 5.3, say, 1 and 2, and construct compositions of pairs located in the corresponding cells by the rule: $R_{12}(x_1, x_2) = R_1(x_1)\times R_2(x_2)$ and $C_{12}(x_1, x_2) = C_1(x_1) + C_2(x_2)$.

Table 5.4. Dominating sequence for the composition of groups1 and 2.

	Number of redundant units of the 1st redundant group						
		0	1	2	…	n	…
Number of redundant units of the 1st redundant group	0	$R_{12}(0,0)$, $C_{12}(0,0)$	$R_{12}(1,0)$, $C_{12}(1,0)$	$R_{12}(2,0)$, $C_{12}(2,0)$	…	$R_{12}(n,0)$, $C_{12}(n,0)$	…
	1	$R_{12}(0,1)$, $C_{12}(0,1)$	$R_{12}(1,1)$, $C_{12}(1,1)$	$R_{12}(2,1)$, $C_{12}(2,1)$	…	$R_{12}(n,1)$, $C_{12}(n,1)$	…
	2	$R_{12}(0,2)$, $C_{12}(0,2)$	$R_{12}(1,2)$, $C_{12}(1,2)$	$R_{12}(2,2)$, $C_{12}(2,2)$	…	$R_{12}(n,2)$, $C_{12}(n,2)$	…
	…	…	…	…	…	…	…
	m	$R_{12}(0,n)$, $C_{12}(0,n)$	$R_{12}(1,m)$, $C_{12}(1,m)$	$R_{12}(2,m)$, $C_{12}(2,m)$	…	$R_{12}(n, m)$, $C_{12}(n, m)$	…
	…	…	…	…	…	…	…

The size of the table (i.e. values m and n) is not fixed a priori. It could be increased if a sequence of dominating pairs $\{R_{12}(x_1, x_2), C_{12}(x_1, x_2)\}$ does not include a desired solution.

As the result, now we have a system with $n - 1$ redundant groups: groups from 3 to N and one new group formed by the described composing of groups 1 and 2.

The procedure continues until one obtains a single composed group that is used in both cases: for solving direct as well as inverse problem of optimal redundancy.

5.2. Numerical example.. For demonstration of the Kettelle's Algorithm, let us consider a simple numerical example with a system of three redundant groups of active units.

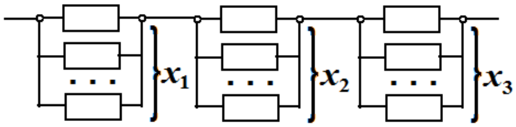


Figure 5.3. Block diagram of the system for the numerical example.

Let $R_j(x_j) = 1 - q_j^{x_j}$ and $C_j(x_j) = c_j x_j$ where $1 \le j \le 3$. Assume that $q_1 = 0.3$, $q_2 = 0.5$, $q_3 = 0.5$, and $c_1=1$, $c_2=3$, $c_3 = 1$.

For the sake of calculating convenience, let us prepare in advance dominated sequences for each redundant group, presented in Table 3. (Notice that for a single group, sequence of pairs "reliability-cost" is always dominating, since each added unit increases simultaneously both the cost and reliability index.)

Table 5.5. Initial dominating sequences for redundant groups 1, 2 and 3.

Group number		Number of redundant units in the group						
		0	1	2	3	4	5	…
1	*R*	0.7000	0.9100	0.9730	0.9919	0.9976	0.9992	…
	C	0	1	2	3	4	5	…
2	*R*	0.5000	0.7500	0.8750	0.9375	0.9688	0.9844	…
	C	0	3	6	8	12	15	…
3	*R*	0.5000	0.7500	0.8750	0.9375	0.9688	0.9844	…
	C	0	1	2	3	4	5	…

It is easy to see that each function *R*(*C*) is concave.

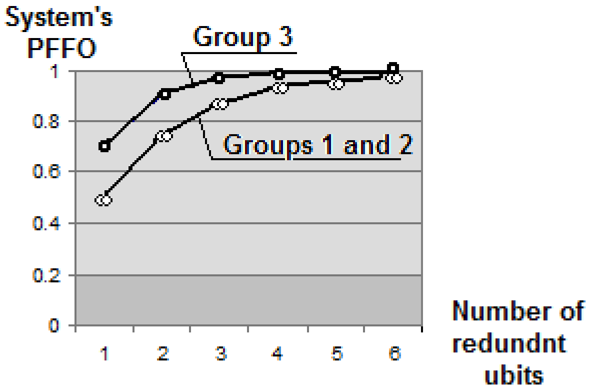


Figure 5.4. Concave shapes of R(C) functions for initial redundant groups.

Next step is construction of the table with various combinations of possible configurations of groups 1 and 2 by the rules $R(x_1, x_2) = (1 - q_1^{x_1}) \times (1 - q_2^{x_2})$ and $C(x_1, x_2) = c_1 x_1 + c_2 x_2$.

Table 5.6. Dominating sequence for the composition of groups1 and 2.

		x_2					
		0	1	2	3	4	5
x_1	0	0.350; 0 ***1***	0.525; 3 ***4***	0.613; 6	0.656; 9	0.678; 12	0.695; 1 5
	1	0.454; 1 ***2***	0.683; 4 ***5***	0.796; 7 ***8***	0.853; 10	0.882; 13	0.896; 16
	2	0.487; 2 ***3***	0.730; 5 ***6***	0.851; 8 ***9***	0.912 1 ***12***	0.942; 14 ***15***	0.958; 17
	3	0.496; 3	0.744; 6 ***7***	0.868; 9 ***10***	0.930; 12 ***13***	0.961; 15 ***16***	0.976; 18 ***19***
	4	0.499; 4	0.748; 7	0.873; 10	0.935; 13	0.966; 16	0.981; 19

				11	***14***	***17***	***20***
	5	0.500; 5	0.749; 8	0.874; 11	0.937; 14	0.968; 17 ***18***	0.984; 20 ***21***
	…	…	…	…	…	…	

Numbers in bold Italic denote the members of dominating pairs by ascending ordering by weights.

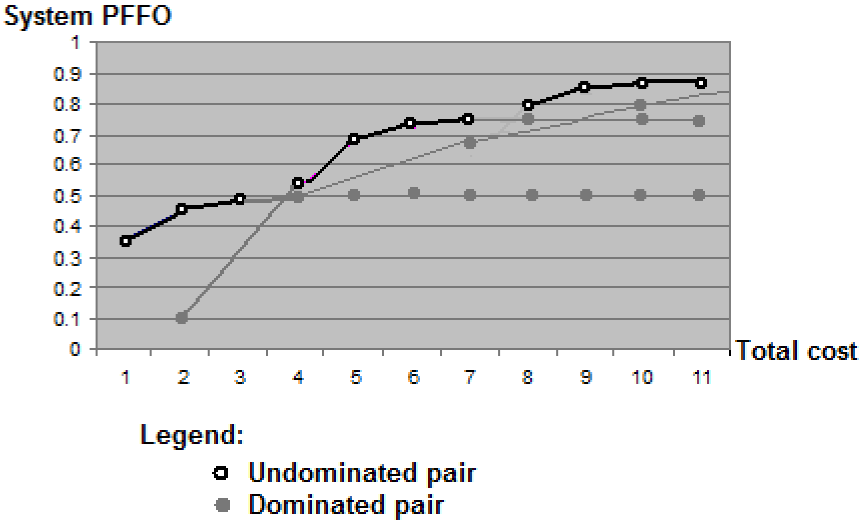


Figure 5.5. Graphical presentation of the data in the upper left corner of Table 5.6.

From Figure 5.5, one can see that the dominating sequence is not strictly concave, though there is some kind of concave envelope that, by the way, very often coincides with solutions obtained by the SDM.

(3) On the basis of Table 5.6, one constructs Table 5.7 containing only dominating reliability-cost pairs. For the illustrative example such a table has the form:

Table 5.7. Beginning of the dominating sequence in Table 6.6.

Number	1	2	3	4	5	6	7	8	9	10	11	…
Domin.pair	0.350; 0	0.454; 1	0.487; 2	0.525; 3	0.683; 4	0.730; 5	0.744; 6	0.796; 7	0.851; 8	0.868; 9	0.873; 10	…
(x_1, x_2)	(0, 0)	(0, 1)	(0, 2)	(0, 3)	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	…

Let us enumerate corresponding pairs of this dominating sequence with the number $x^{(1)}$.

(4) Now we have a system consisting of two redundant groups: group 3 (data are on the lower lines of Table 5.5) and the newly composed group (data are in Table 5.7). On the basis of these data let us combine the final group for the considering system.

Table 5.8. Final data for solving the optimal redundancy problems for the illustrative example.

		Number of dominating variantof pairs { $x^{(1)}$= (x_1, x_2)}									
		1	2	3	4	5	6	7	8	9	. . .
x_3	0	0.175; 0 ***1***	0.227; 1	0.243; 2	0.262; 3	0.341; 4	0.365; 5	0.372; 6	0.398; 7	0.425; 8	. . .
	1	0.263; 1 ***2***	0.341; 2 ***3***	0.365; 3	0.393; 4	0.512; 5 *6*	0.546; 6	0.556; 7	0.596; 8	0.638; 9	. . .
	2	0.306; 2	0.398; 3 ***4***	0.425; 4	0.458; 5	0.597; 6 ***7***	0.639; 7	0.650; 8	0.696; 9	0.745; 10	. . .
	3	0.330; 3	0.427; 4 ***5***	0.456; 5	0.492; 6	0.640; 7 ***8***	0.685; 8 ***9***	0.705; 9	0.746; 10 *11*	0.798; 11 *12*	. . .
	4	0.340; 4	0.441; 5	0.472; 6	0.510; 7	0.661; 8	0.707; 9 ***10***	0.720; 10	0.772; 11	0.825; 12 *13*	. . .
	5	0.343; 5	0.7448; 6	0.479; 7	0.516; 8	0.672; 9	0.719; 10	0.732 11	0.784; 12	0.837; 13	. . .
. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .

Table 5.9. Final dominating sequence for the system

R	0.175	0.263	0.341	0.398	0.427	0.512	0.597	0.640	0.685	0.707	0.746	0.798	0.825	…
C	0	1	2	3	4	5	6	7	8	9	10	11	12	…

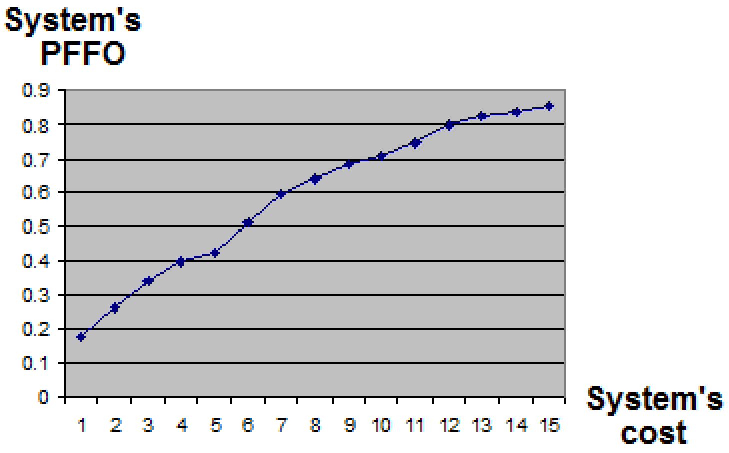


Figure 5.6. Final dominating sequence for the system.

5.2.3. Solving the direct and inverse problems of optimal redundancy.

Using Table 5.7, it is easy to get solutions for the both – direct as well as inverse – problem of optimal redundancy. For instance, if one needs to find the best redundant units allocation to satisfy the requirement R ≥ 0.8, then from Table7 we find that the solution for corresponding value (*R*=0.825) is in cell ($x^{(1)}$ =9, x_3 = 4). The cost of redundant units in this case is 12 In turn, for $x^{(1)}$ =9 one finds that this corresponds to x_2=1 and x_2=2.

Thus, the solution of inverse problem is (x_1=2, x_2=2, x_3=3).

If there is a limitation on the redundant units total cost equal to 10, then from the

Table 5.7 one finds that corresponding maximum value of PFFO is 0.746. This solution corresponds to a cell ($x^{(1)}$ =8, x_3 = 3). In turn, for $x^{(1)}$ =8 one finds that this corresponds to x_1=1 and x_2=2.

Thus, the solution of inverse problem is (x_1=1, x_2=2, x_3=3).

Remark. There are two ways of choosing redundant groups for composing a dominating sequence (see Figure 5.7).

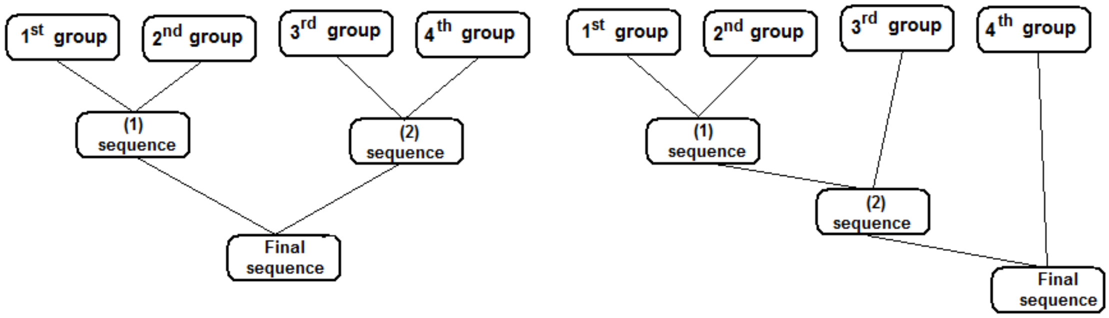


Figure 5.7. Two types of choosing redundant group for composing a dominating sequence

For computer solution, both types are equivalent. However if one needs to make calculations by hand, then the dichotomous way gives a substantial decrease in calculations.

* * *

Bibliography to Chapter 5.
(books a distinguished by bold font)

1957. Bellman, R. Dynamic Programming. Princeton University Press.

1962. Bellman, R. E., and Dreyfus, S. E. Applied Dynamic Programming. Princeton University Press.

1962. Kettelle, J. D. , Jr. Least-coast allocation of reliability investment. *Operations Research*, No. 2.

1966. Li, J. A bound dynamic programming for solving reliability redundancy optimization . Microelectronics Reliability, No. 10.

1969. Ushakov, I.A. Method of Solving Optimal Redundancy Problems under Constraints (in Russian). Sovetskoe Radio, Moscow .

1971. Misra , K.B. Dynamic programming formulation of the redundancy allocation problem. International Journal of Mathematical Education in Science and Technology, No.3.

2005. Yalaoui, A., Chatelet, E., and Chu, C. A new dynamic programming method for reliability and redundancy allocation in a parallel-series system. IEEE Transactions on Reliability, No. 2.

6.UNIVERSAL GENERATING FUNCTIONS

The Method of Universal Generating Functions (U-functions), introduced in [Ushakov, 1986], actually represents a generalization of the Kettelle's Algorithm. This method suggests a transparent and convenient method of computerized solutions of various enumeration problems, in particular, the optimal redundancy problem.

6.1. Generating function

First, let us refresh our memory concerning generating functions. This is a very convenient tool widely used in the probability theory for finding joint distributions of several discrete random variables. Generating function is defined as

$$\varphi(z) = \sum_{k \in G_k} p_k z^k \tag{6.1}$$

where p_k is the probability that discrete random variable *X* takes value *k* and G_k is the distribution function domain. In the optimal redundancy problems, in principle, $G_k = [0, \infty)$, though any practical task has its own limitations on the largest value of *k*.

Consider two non-negative discrete random values X_1 and X_2 with distributions

$$\begin{cases} P\{X^{(1)} = 1\} = p_1^{(1)}, \\ P\{X^{(1)} = 2\} = p_2^{(1)} \\ \qquad \dots \\ P\{X^{(1)} = n_1\} = p_{n_1}^{(1)}, \end{cases}$$

and

$$\begin{cases} P\{X^{(2)} = 1\} = p_1^{(2)}, \\ P\{X^{(2)} = 2\} = p_2^{(2)}, \\ \qquad \dots \\ P\{X^{(2)} = n_2\} = p_{n_2}^{(2)}. \end{cases}$$

correspondingly, where n_1 and n_2 are numbers of discrete values of each type.

For finding probability of random variable $X = X^{(1)} + X^{(2)}$, one should enumerate all possible pairs of $X^{(1)}$ and $X^{(2)}$ that give in sum value *k* and add corresponding probabilities:

$X^{(1)} = 0$, $X^{(2)} = k$, with probability $p_0^{(1)} \cdot p_k^{(2)}$

$X^{(1)} = 1$, $X^{(2)} = k - 1$, with probability $p_1^{(1)} \cdot p_{k-1}^{(2)}$

$X^{(1)} = 2,\ X^{(2)} = k - 2$, with probability $p_2^{(1)} \cdot p_{k-2}^{(2)}$

…

$X^{(1)} = k,\ X^{(2)} = 0$, with probability $p_k^{(1)} \cdot p_0^{(2)}$.

Thus the probability of interest is equal to

$$P\{X = k\} = \sum_{0 \le j \le k} p_j^{(1)} \cdot p_{k-j}^{(2)} = \sum_{0 \le j \le k} p_{k-j}^{(1)} \cdot p_j^{(2)} \tag{6.2}$$

One can see that there is a comvolution transform. It is clear that the same result will be obtained if one takes a polynomial

$$\varphi\ (z) = \varphi^{(1)}(z) \cdot \varphi^{(2)}(z) \tag{6.3}$$

and, after combining alike terms of expansions, finds the coefficient at z^k.

6.2. Universal GF (U-function)

One sees that algebraic argument “z” was introduced only for convenience: everybody knows that polynomials multiplication means product of coefficients and sum of powers. Such presentation helps one to obtain a distribution of the convolution of discrete r.v.’s. However, if random variables should be expose transformation different from convolution? For instance, if these random variables are arguments of some function?

Let us use habitual form of presentation, using symbol “$\otimes$” instead of “$\prod$” just to underline that this is not an ordinary product of two GFs but special transform:

$$\varphi(z) = \varphi^{(1)}(z) \underset{f}{\otimes} \varphi^{(2)}(z) = \left(\sum_{1 \le i \le n_1} p_i^{(1)} z^{X_i^{(1)}} \right) \underset{f}{\otimes} \left(\sum_{1 \le j \le n_2} p_j^{(2)} z^{X_j^{(2)}} \right) = \sum_{\substack{1 \le i \le n_1 \\ 1 \le j \le n_2}} p_i^{(1)} p_j^{(2)} z^{f(X_i^{(1)}, X_j^{(2)})} \tag{6.4}$$

Subscript “f” in $\underset{f}{\otimes}$ means that some specific operation f will be undertaken over values X. It is clear that in case of “pure” GF function is operation of summation.

In general case, using polynomial form of GF is inconvenient and even impossible. For moving further, let us introduce some terms. We used to say that a system consists of units which are physical objects characterized by its parameters: reliability, cost, weight, etc. So, we can consider each unit as a multiplet of its parameter. Relaibility of each unit can be improved by using redundancy or by replacing with more effective unit. In other words, on a design stage engineer feals with a “string” of possible multiplets characterizing various variants of a considered unit.

Consider a series system of two units. Let unit-1 and unit-2 are characterized by strings

$$S_1 = \{M_1^{(1)}, M_2^{(1)}, ..., M_{n_1}^{(1)}\}$$

and

$$S_2 = \{M_1^{(2)}, M_2^{(2)}, ..., M_{n_2}^{(2)}\}$$

Each multiplet is a set of parameters $M_j^{(k)} = \{\alpha_{1j}^{(k)}, \alpha_{2j}^{(k)}, ..., \alpha_N^{(k)}\}$ where N is the number of parameters in each multiplet.

"Interaction" of these two strings is an analogue of the Cartezian product whose memberts fill the cells of the following table:

Table 6.1.

	$M_1^{(1)}$	$M_2^{(1)}$	…	$M_{n_1}^{(1)}$
$M_1^{(2)}$	$M_1^{(1)} \otimes M_1^{(2)}$	$M_2^{(1)} \otimes M_1^{(2)}$	…	$M_{n_1}^{(1)} \otimes M_1^{(2)}$
$M_2^{(2)}$	$M_1^{(1)} \otimes M_2^{(2)}$	$M_2^{(1)} \otimes M_2^{(2)}$	…	$M_{n_1}^{(1)} \otimes M_2^{(2)}$
…	…	…	…	…
$M_{n_2}^{(2)}$	$M_1^{(1)} \otimes M_{n_2}^{(2)}$	$M_2^{(1)} \otimes M_{n_2}^{(2)}$	…	$M_{n_1}^{(1)} \otimes M_{n_2}^{(2)}$

Interaction of multiplets consists of iteraction of their similar parameters, for instance,

$$M_j^{(k)} \otimes M_i^{(h)} = \{(\alpha_{1j}^{(k)} \underset{f_1}{\otimes} \alpha_{1i}^{(h)}), (\alpha_{2j}^{(k)} \underset{f_2}{\otimes} \alpha_{2i}^{(h)}), ..., (\alpha_{Nj}^{(k)} \underset{f_N}{\otimes} \alpha_{Ni}^{(h)})\} \tag{6.5}$$

Operator $\otimes$, as well as each operator $\underset{f_s}{\otimes}$, in most natural practical cases possesses the commutativity property, i.e.

$$\underset{f}{\otimes}(a, b) = \underset{f}{\otimes}(b, a), \tag{6.6}$$

and the associativity property, i.e.

$$\underset{f}{\otimes}(a, b, c) = \underset{f}{\otimes}(a \underset{f}{\otimes}(b, c)) = \underset{f}{\otimes}((a \underset{f}{\otimes} b), c). \tag{6.7}$$

Of course, operator $\underset{f_s}{\otimes}$ depends on the physical nature of parameter α_s and the type of structure, i.e. series or parallel.

Table 6. 2.

Type of parameter	Type of structure	Result of interaction
A) α is unit's PFFO	series	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \alpha_{Aj}^{(k)} \times \alpha_{Ai}^{(h)}$
	parallel	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = 1 - (1 - \alpha_{Aj}^{(k)}) \times (1 - \alpha_{Ai}^{(h)})$
B) α is number of units in parallel	series	$\alpha_{Bj}^{(k)} \underset{f}{\otimes} \alpha_{Bi}^{(h)} = (\alpha_{Bj}^{(k)}; B_{Ri}^{(h)})$
	parallel	$\alpha_{Bj}^{(k)} \underset{f}{\otimes} \alpha_{Bi}^{(h)} = (\alpha_{Bj}^{(k)}; B_{Ri}^{(h)})$
C) α is unit's cost (weight)	series	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \alpha_{Aj}^{(k)} + \alpha_{Ai}^{(h)}$
	parallel	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \alpha_{Aj}^{(k)} + \alpha_{Ai}^{(h)}$
D) α is unit's ohmic resistance	series	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \alpha_{Aj}^{(k)} + \alpha_{Ai}^{(h)}$
	parallel	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \left[(\alpha_{Aj}^{(k)})^{-1} + (\alpha_{Ai}^{(h)})^{-1}\right]^{-1}$
E) α is unit's capacitance	series	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \left[(\alpha_{Aj}^{(k)})^{-1} + (\alpha_{Ai}^{(h)})^{-1}\right]^{-1}$
	parallel	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \alpha_{Aj}^{(k)} + \alpha_{Ai}^{(h)}$
F) α is pipeline unit's capacitance	series	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \min\left\{\alpha_{Aj}^{(k)}, \alpha_{Ai}^{(h)}\right\}$
	parallel	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \alpha_{Aj}^{(k)} + \alpha_{Ai}^{(h)}$
G) α is unit's random time to failure	series	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \min\left\{\alpha_{Aj}^{(k)}, \alpha_{Ai}^{(h)}\right\}$
	parallel	$\alpha_{Aj}^{(k)} \underset{f}{\otimes} \alpha_{Ai}^{(h)} = \max\left\{\alpha_{Aj}^{(k)}, \alpha_{Ai}^{(h)}\right\}$

In the problem of optimal redundancy, one deals with triplet of type "*Probability-Cost-Number of units*" for each redundant group: $M_j = \{\alpha_{1j}, \alpha_{2j}, \alpha_{3j}\}$. If there is a system of *n* series subsystems (single elements or redundant groups) , one has to use a procedure almost completely coincided with the procedure of compiling the dominating sequences at the Kettelle's algorithm. In other words, the problem reduces to the constructing a single "equivalent unit" which possesses the entire system's properties. There are two possible ways of "convolving" the system into a single "equivalent unit": dichotomous and sequential. We will demonstrate these two possible procedures on an exmple of a series system of four subsystems.

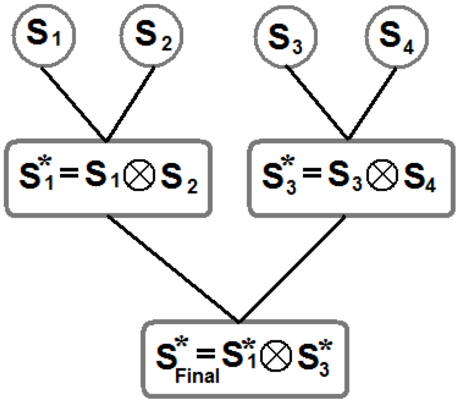

S_1
S_2
S_3
S_4
$S_1^* = S_1 \otimes S_2$
$S_3^* = S_3 \otimes S_4$
$S_{Final}^* = S_1^* \otimes S_3^*$

Figure 6. 1 Dichotomous scheme of compiling the equivalent unit.

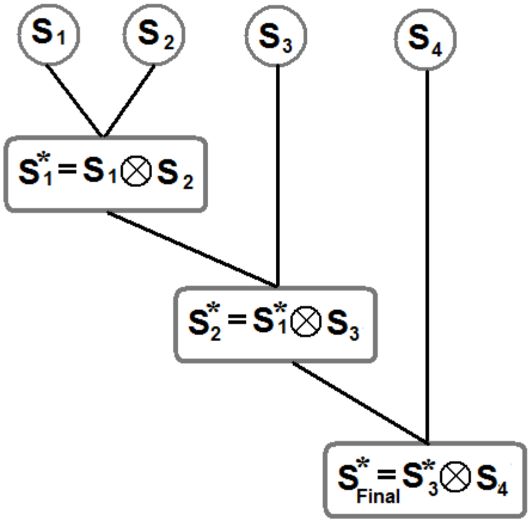

S_1
S_2
S_3
S_4
$S_1^* = S_1 \otimes S_2$
$S_2^* = S_1^* \otimes S_3$
$S_{Final}^* = S_3^* \otimes S_4$

Figure 6.2. Dichotomous scheme of compiling the equivalent unit.

Numerical example. Consider a series system of four units with parameters given in the table below. Assume that "hot" redundancy of each unit is used for the system reliability improvement.

Table 6.3. System unit parameters

	Unit-1	Unit-2	Unit-3	Unit-4
PFFO	0.6	0.6	0.7	0.7
Cost	3	1.5	2	1.2

Let us solve two problems of optimal redundancy:

(a) Find the optimal allocation of redundant units to reach required PFFO level of 0.95;

(b) Find the optimal allocation of redundant units to reach maximum possible PFO level under condition that the total cost of the system does not exceed 30 cost units.

In this case, each unit is characterized by the following strings of triplets (Cost, PFFO, Number):

S_1 = [{3; 0.6; 1}, ... , {12; 0.9744; 4}, {15; 0.9898; 5}, {18; 0.9959; 6}, {21; 0.9984; 7}, {24; 0.9993; 8}, {27; 0.9997; 9}, ...]

S_2 = [{1.5; 0.6; 1}, ... , {6; 0.9744; 4}, {7.5; 0.9898; 5}, {9; 0.9959; 6}, {10.5; 0.9984; 7}, {12; 0.9993; 8}, {13.5; 0.9997; 9}, ...]

S_3 = [{2; 0.7; 1}, ... , {6; 0.9730; 3}, {8; 0.9919; 4}, {10; 0.9976; 5}, {12; 0.9993; 6}, {14; 0.9998; 7}, {16; 0.9999; 8}, ...]

S_4 = [{1.2; 0.7; 1}, ... , {3.6; 0.9730; 3}, {4.8; 0.9919; 4}, {6; 0.9976; 5}, {7.2; 0.9993; 6}, {8.4; 0.9998; 7}, {9.6; 0.9999; 8}, ...].

Let us apply dichotomous scheme of compikling equivalent unit, and, first, consider the subsystem consisting of Unit-1 and Unit -2. We omit all intermediate calculations performed with help of a simple Excel program.

Table 6.4. Triplets belonging to S_1^* .

Cost	**PFFO**	**$X=(x_1,x_2)$**
…	…	…
18	0.9495	(4,4)
19.5	0.9644	(4,5)
21	0.9704	(4,6)
21	0.9644	(5,4)
22.5	0.9796	(5,5)
22.5	0.9728	(4,7)
24	0.9857	(5,6)
24	0.9704	(6,4)
24	0.9738	(4,8)
25.5	0.9881	(5,7)
25.5	0.9857	(6,5)
25.5	0.9741	(4,9)
27	0.9728	(7,4)
27	0.9918	(6,6)
27	0.9891	(5,8)
28.5	0.9881	(7,5)
28.5	0.9943	(6,7)
28.5	0.9895	(5,9)
30	0.9738	(8,4)
30	0.9943	(7,6)
30	0.9953	(6,8)
…	…	…

In this table, as well as in next tables below, shadowed are those triplets which are dominated by dominating ones.

In the same manner construct S_2^*.

Table 6.5. Triplets belonging to S_2^*.

Cost	**PFFO**	**$X=(x_3,x_4)$**
…	…	…
9.6	0.9467	(3,3)
10.8	0.9651	(3,4)
11.6	0.9651	(4,3)
12	0.9706	(3,5)
12.8	0.9839	(4,4)
13.2	0.9723	(3,6)
13.6	0.9706	(5,3)
14	0.9895	(4,5)
14.4	0.9728	(3,7)
14.8	0.9895	(5,4)
15.2	0.9912	(4,6)
15.6	0.9723	(6,3)
15.6	0.9729	(3,8)
16	0.9951	(5,5)
…	…	…

Now on the basis of S_1^* and S_2^*, one constructs the final string for the equivalent unit. The result is given in the table below.

Table 6.6. Resulting string of triplets for the equivalent unit.

Cost	**PFFO**	**$X=(x_3,x_4,x_3,x_4)$**
…	…	…
27.6	0.8989	(4,4,2,3)
28.8	0.9164	(4,4,4,3)
29.1	0.9164	(4,5,2,3)
30.0	0.9216	(4,4,5,3)
30.3	0.9307	(4,5,4,3)
30.6	0.9187	(3,6,2,3)
30.8	0.9342	(4,4,4,8)
31.5	0.9360	(4,5,5,3)
31.8	0.9365	(3,6,4,3)
32.0	0.9395	(4,4,5,4)
32.1	0.9274	(5,5,2,3)
32.3	0.9489	(4,5,4,8)
33.0	0.9419	(3,6,5,3)
33.2	0.9411	(4,4,6,4)
33.3	0.9454	(5,5,4,3)
33.5	0.9543	(4,5,5,4)
33.6	0.9389	(5,6,2,3)
33.8	0.9548	(3,6,4,8)

…   …   …

One can notice that the final string in this particular case completely coincides with the final dominating sequence obtained by the Kettelle's Algorithm: the only difference is that we kept "the track of solving process" and have the resulting solution immediately from the table. (Frankly speaking, the Kettele's Algorithm could be easily modified to get the same property of the final solution.)

Solutions of the problems above can be easily found from the last table. First time PFFO exceed level of 0.95 when X=(4,5,5,4) and the corresponding system cost is 33.5 cost units. The second task has solution X=(4,4,5,3) with the cost equal exactly 30 cost units. FFO for this case is equal to 0.9216.

In the conclusion of this chapter, let us notice that the *U*-function method is very constructive not only for solving optimal redundancy problem, but also for a number of other problems, particularly associated with multi-state systems analysis.

Bibliography to Chapter 6

(books are distinguished by bold font)

1986. Ushakov, I.A. A Universal Generating Function (in Russian). Engeneering Cybernetics, No. 5.

1987. Ushakov, I.A. Solution of multi-criteria discrete optimization problems using a universal generating function. Soviet Journal of Computer and System Sciences , No. 5.

1987. Ushakov, I. A universal generating function. Soviet Journal Computer Systems Science, No.3.

1987. Ushakov, I.A. Optimal standby problem and a universal generating function. Soviet Journal Computer and System Science, No.4.

1988. Ushakov, I.A. Solving of optimal redundancy problem by means of a generalized generating function. Elektronische Informationsverarbeitung und Kybernetik, No.4-5

2000. Ushakov, I.A. The method of generating sequences. European Journal of Operational Research, No. 2.

2000. Ushakov, I.A. The Method of Generating Sequences. *European Journal of Operational Research*, Vol. 125/2

2005. Levitin, G. The Universal Generating Function in Reliability Analysis and Optimisation. Springer-Verlag.

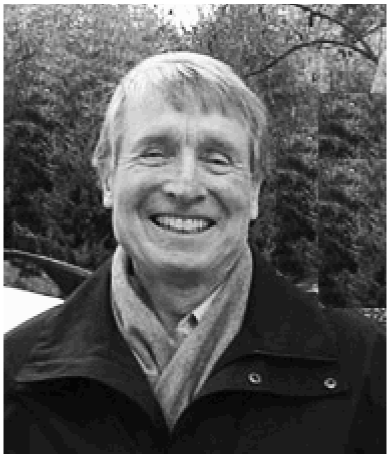
2008. Chakravarty, S.,and Ushakov, I.A. Object Oriented Commonalities in Universal Generating Function for Reliability and in C++, Reliability and Risk Analysis: Theory & Applications, No. 10.

7.GENETIC ALGORITHMS

7.1. Introductory

Computer simulations of evolution started in the mid-1950s. A Norwegian-Italian mathematician Nils Aall Barricelli (1912-1993), who had been working at the Institute for Advanced Study in Princeton, published his first paper on the subject. Then a series of works have been published in 1960-70s.

Genetic algorithms in particular became popular through the work of John Holland in the early 1970s., and particularly his book “Adaptation in Natural and Artificial Systems” (1975).



John Henry 'Dutchy' Holland
(born in 1929)

American scientist and Professor of Psychology and Professor of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor. He is a pioneer in complex system and nonlinear science. He is known as the father of Genetic Algorithms. In 1975 he wrote his book on genetic algorithms, "Adaptation in Natural and Artificial Systems".

Holland wrote: "A Genetic Algorithm is a method of problem analysis based on Darwin's theory of natural selection. It starts with an initial population of individual nodes, each with randomly generated characteristics. Each is evaluated by some method to see which ones are more successful. These successful ones are then merged into one "child" that has a combination of traits of the parents’ characteristics."

In recent years, many studies on reliability optimization use a universal optimization approach based on metaheuristics. Genetic algorithms are considered as a particular class of evolutionary algorithms that use techniques inspired by Darwin’s evolution theory in biology that includes such components as inheritance, mutation, selection, and crossover (recombination).

These metaheuristics hardly depend on the specific nature of the problem that is being solved and, therefore, can be easily applied to solve a wide range of optimization problems. The metaheuristics are based on artificial reasoning rather than on classical mathematical programming. An important advantage of these methods is that they do not require any prior information and are based on collection of current data obtained during the randomized search process. These data are substantially used for directing the search.

Genetic algorithms are implemented as a computer simulation in which a population of abstract items (called “chromosomes” or “the genotype of the genome”) represent “candidate solutions” (called individuals, creatures, or phenotypes) systematically lead toward better solutions.

The GAs have the following advantages in comparison with traditional methods:

- they can be easily implemented and adapted;
- they usually converge rapidly on solutions of good quality;
- they can easily handle constrained optimization problems.

A genetic algorithm requires strong definition of two things:

a genetic representation of the solution domain,

a fitness function to evaluate the solution domain.

The fitness function is defined over the genetic representation and measures the quality of the presented solution. The fitness function always depends on the problem nature. In some problems, it is impossible to define the fitness expression, so one needs to use interactive procedures based on expert’s opinion.

As soon as we have the genetic representation and the properly defined fitness function, GA proceeds to initialize a population of solutions randomly.

Genetic algorithm includes the following main phases.

Initialization

A number of individual solutions is generated at random to form an initial population. The population size depends on the nature of the problem, but typically contains hundreds or thousands of possible solutions. The population is generated to be able to cover the entire range of possible solutions (the search space). Occasionally, some solutions may be "seeded" in areas where actual optimal solution is located.

This initial population of solutions is undertaken to improve the procedure through repetitive application of selection, reproduction, mutation, and crossover operators.

Selection

Obtained individual solutions are selected through a special process using a fitness function that allows ordering the solutions by specified quality measure. These selection methods rate the fitness of each solution and preferentially select the best solutions.

Reproduction

The next step is generating the next generation of solutions from those selected through genetic operators: crossover (recombination), and mutation.

Each new solution is produced by a pair of "parent" solutions that selected for “breeding”. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated.

Termination

This process described above is repeated until a termination condition has been reached. Common terminating conditions are:

the predetermined number of produced generations has been reached,
or a satisfactory fitness level has been reached for the population.

7.2. Structure of Steady-state Genetic Algorithms[3]

The steady-state GA (see Figure 7.1) proceeds as follows: an initial population of solutions is generated randomly or heuristically. Within this population, new solutions are obtained during the genetic cycle by using the crossover operator. This operator produces an offspring from a randomly selected pair of parent solutions that are selected with a probability proportional to their relative fitness. The newly obtained offspring undergoes mutation with the probability p_{mut}.

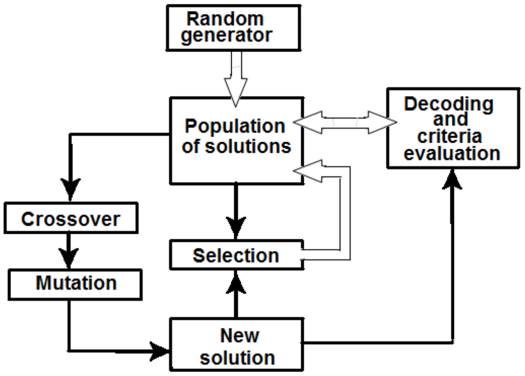


Figure 7.2. Structure of a steady-state GA

Each new solution is decoded and its fitness function value is estimated. These values are used for a selection procedure that determines what is better: the newly obtained solution or the worst solution in the population. The better solution joins the population, while the current one is discarded. If the solution population contains a pair of equivalent items, then either of them is eliminated and the population size decreases. The stopping rule is when the number of new solutions reaches some level N_{rep}, or when the number of remained solutions in the population after excluding reaches a specified level. After this, the new genetic cycle begins: a new population of randomly constructed solutions is generated and the process continues. The whole optimization process terminates when its specified termination condition is satisfied. This condition can be specified in the same way as in a generational GA.

The steady-state GA can be presented in the following pseudo-code format.

```
begin STEADY STATE GA
 Initialize population $\Pi$
 Evaluate population $\Pi$ {compute fitness values}
 while GA termination criterion is not satisfied do
    {GENETIC CYCLE}
    while genetic cycle termination criterion is not satisfied do
             Select at random Parent Solutions $S_1$, $S_2$ from $\Pi$
```

[3] Material for this section is presented by G. Levitin.

```
                Crossover: $(S_1, S_2) \rightarrow S_O$ {offspring}
                Mutate offspring $S_O \rightarrow S^*_O$ with probability $p_{mut}$
                Evaluate $S^*_O$
                Replace $S_W$ {the worst solution in $\Pi$ with $S^*_O$ } if $S^*_O$ is
                better than $S_W$
                Eliminate identical solutions in $\Pi$
        end while
        Replenish $\Pi$ with new randomly generated solutions
    end while
  end GA
```

7.3. Related techniques[4]

Below is given a list (in alphabetic order) of a umber of techniques “genetically” close to Genetic Algorithm:

Ant colony optimization (ACO) uses many ants (or agents) to traverse the solution space and find locally productive areas.

Bacteriologic Algorithms (BA) inspired by evolutionary ecology and, more particularly, bacteriologic adaptation.

Cross-entropy method (CE) generates candidates solutions via a parameterized probability distribution.

Evolution strategies (ES) evolve individuals by means of mutation and intermediate and discrete recombination.

Evolutionary programming (EP) involves populations of solutions with primarily mutation and selection and arbitrary representations.

Extremal optimization (EO) evolves a single solution and makes local modifications to the worst components.

Genetic programming (GP) is a technique, in which computer programs, rather than function parameters, are optimized.

Deluge algorithm (GD) is a generic algorithm similar in many ways to the hill-climbing and simulated annealing algorithms.

Grouping Genetic Algorithm (GGA) is an evolution of the GA where the focus is shifted from individual items, like in classical GAs, to groups or subset of items.

Harmony search (HS) is an algorithm mimicking musician’s behaviors in improvisation process.

Immune optimization algorithm (IOA) is based on both the concept of Pareto optimality and simple interactive metaphors between antibody population and multiple antigens.

Interactive evolutionary algorithms (IEA) are evolutionary algorithms that use human evaluation when it is hard to design a computational fitness function.

Mimetic algorithm (MA) is a relatively new evolutionary method where local search

[4] This section is based partly on http://en.wikipedia.org/wiki/Genetic_algorithm.

is applied during the evolutionary cycle.

Particle swarm optimization (PSO) is an algorithm to find a solution to an optimization problem in a search space, or model and predict social behavior in the presence of objectives.

Simulated annealing (SA) is a related global optimization technique that traverses the search space by testing random mutations on an individual solution.

Taboo search (TS) is similar to Simulated Annealing in that both traverse the solution space by testing mutations of an individual solution.

Bibliography to Chapter 7

1966. Coit, D.W., and Smith, A.E. Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Transactions on Reliability, No. 45

1966. Coit, D.W., Smith, A.E. and Tate, D.M. Adaptive penalty methods for genetic optimization of constrained combinatorial problems, INFORMS Journal of Computing, No. 8 .

1995. Kumar, A., Pathak, R., and Gupta, Y. Genetic algorithm-based reliability optimization for computer network expansion, IEEE Transaction on Reliability, No. 1.

1995. Kumar, A., Pathak, R., Gupta, and Parsaei, H. A genetic algorithm for distributed system topology design. Computers and Industrial Engineering, No 3.

1995. Painton, L., and Campbell, J. Genetic algorithm in optimization of system reliability, IEEE Transactions on Reliability, No. 2.

1996. Coit, D.W., and Smith, A.E. Reliability optimization of series-parallel systems using genetic algorithm. IEEE Transactions on. Reliability, No. 2.

1996. Coit, D.W., and Smith, A.E. Solving the redundancy allocation problem using a combined neural network/genetic algorithm approach. Computers & OR, No.6.

1996.Coit, D.W., and Smith, A.E. Penalty guided genetic search for reliability design optimization. Computers and Industrial Engineering, No. 30

1997. Dengiz, B., Altiparmak, F., and Smith, A. Local search genetic algorithm for optimal design of reliable networks, IEEE Transaction on evolutionary computation, No. 3.

1997. Dengiz, B., Altiparmak, F., and Smith, A. Efficient optimization of all-terminal reliable networks,using an evolutionary approach, IEEE Transaction on Reliability, vol. 46 No. 1.

1997. Gen, M., and Cheng, R. Genetic Algorithms and Engineering Design, , John Wiley & Sons, New York.

1997. Gen, M., and Kim, J. GA-based reliability design: state-of-the-art survey. Computers and Industrial Engineering, No 1/2.

1997. Ramachandran, V., Sivakumar, V., and Sathiyanarayanan, K. Genetics based redundancy optimization, Microelectronics and Reliability, No 4.

1997. Shelokar, P.S., Jayaraman, V. K., and Kulkarni, B. D. Ant algorithm for single and multi-objective reliability optimization problems. Quality and Reliability Engineering International, No. 6

1998. Coit, D.W., and Smith, A.E.) Redundancy allocation to maximize a lower percentile of the system time-to-failure distribution. IEEE Transactions on. Reliability, No. 1.

1998. Hsieh, Y., Chen, T., and Bricker, D. Genetic algorithms for reliability design problems. Microelectronics and Reliability, No 10.

1998. Taguchi, T., Yokota, T., and Gen, M. Reliability optimal design problem with interval coefficients using hybrid genetic algorithms. Computers and Industrial Engineering, No. 1/2.

1999. Yang, J., Hwang, M., Sung, T., and Jin, Y. (1999) Application of genetic algorithm for reliability allocation in nuclear power plant, Reliability Engineering & System Safety, No. 3.

2000. Levitin, G., Kalyuzhny, A., Shenkman, A., and Chertkov, M. Optimal capacitor allocation in distribution systems using a genetic algorithm and a fast energy loss computation technique. IEEE Transactions on Power Delivery, No. 2.

2003. Levitin G., Dai,Y., Xie, M., and Poh, K. L. Optimizing survivability of multi-state systems with multi-level protection by multi-processor genetic algorithm. Reliability Engineering & System Safety, No. 2.

2004. Liang, Y., and Smith, A. An ant colony optimization algorithm for the redundancy allocation problem, IEEE Transactions on Reliability, No. 3.

2005. Coit, D.W., and Baheranwala, F. Solution of stochastic multi-objective system reliability design problems using genetic algorithms. In Advances in Safety and Reliability. Taylor & Francis Group.

2005. Levitin, G. Genetic algorithms in reliability engineering. Reliability Engineering & System Safety, No.2.

2005. Marseguerra, M., Zio, E., and Podofillini, L. Multi-objective spare part allocation by means of genetic algorithms and Monte Carlo simulation, Reliability Engineering & System Safety, No. 87.

2006. Gupta, R., and Agarwal, M. Penalty guided genetic search for redundancy optimization in multi-state series-parallel power system. Journal of Combinatorial Optimization, No. 3.

2006. Levitin, G. (Ed.) Computational Intelligence in Reliability Engineering. Evolutionary Techniques in Reliability Analysis and Optimization, Series: Studies in Computational Intelligence, Vol. 39, Springer-Verlag.

2006. Levitin, G. (Ed.) Computational Intelligence in Reliability Engineering. New Metaheuristics, Neural and Fuzzy Techniques in Reliability. Series: Studies in Computational Intelligence, Vol. 40, Springer-Verlag .

2006. Levitin, G. (Ed.). Computational Intelligence in Reliability Engineering. New Metaheuristics, Neural and Fuzzy Techniques in Reliability. Series: Studies in Computational Intelligence. Springer-Verlag.

2006. Levitin, G. Genetic Algorithms in Reliability Engineering. *Reliability Engineering and System Safety*, 91(9).

2006. Konak, A., Coit, D., Smith. A. Multi-objective optimization using genetic algorithms: a tutorial. *Reliability Engineering and System Safety*, 91(9).

2006. Parkinson, D. Robust design employing a genetic algorithm. Quality and Reliability Engineering International, No. 3.

2007. Levitin, G. Intelligence in Reliability Engineering: New Metaheuristics, Neural and Fuzzy Techniques in Reliability. Springer-Verlag.

2008. Taboada, H.A., Espiritu, J.F., and Coit, D.W. A multi-objective multi-state genetic algorithm for system reliability optimization design problems. IEEE Transactions on Reliability, No. 1.

8.MONTE CARLO SIMULATION

8.1. Introductory remarks

Very often a reliability goal function cannot be expressed in convenient analytical form that makes even calculation of reliability decrements practically impossible. For instance, such situations arise when system units are mutually dependent or their reliability simultaneously depends on some common to all environmental factors (temperature, mechanical impacts, etc.). In these cases, the Monte Carlo simulation is usually used for reliability indices calculation. However, the problem arises: How to use the Monte Carlo simulation for optimization?

Roughly speaking, the idea is in observing the process of the spare unit expenditure (replacement of failed units) until specified restrictions allow one to do so. This may be a simulation process or an observation of the real deployment of the system. After the stopping moment, we start another realization of simulation process or observation of the real data. When the appropriate statistical data are collected, the process of finding optimal solution starts.

Avoiding a formal description of the algorithm, let us demonstrate it on numerical examples which will make the idea of the method and its specific technique clearer.

8.2. Formulation of optimal redundancy problems in statistical terms

Standard methods do not give a solution if the goal function is the mean time to failure

$$T(\mathrm{x}) \;=\; \int_0^\infty \prod_{1\le i\le n} R_i\left(t \middle| x_i\right) dt \qquad (8.1)$$

or if units are dependent, for instance, via a vector of some external factors *g* (temperature, humidity, vibration, etc.):

$$R(\mathrm{x}) \;=\; \int_{g\in G} R(x_1, x_2, \dots x_n|\mathrm{g})dF(\mathrm{g}) \qquad (8.2)$$

where $F(\boldsymbol{g})$ is the d.f. of some external parameter $\boldsymbol{g}$ and G is its domain.

8.3.Algorithm for Trajectory Generation

For solution of the formulated problems, we need to have data obtained from real experiment (or system deployment) or from Monte Carlo simulation of the system model. Though this procedure is routine, we will briefly describe it for the presentation closeness. The procedure is as follows.

Consider a series system of *n* units. (For simplicity of explanation of the algorithm, we will assume that the units are independent. However, everything described below can be easily extended to the general case: it will effect only a mechanism of random sequences generation.)

Let us consider the process of spare units expenditure as the process of changing the system states and the total cost of spare units at sequential replacement moments. After failure each unit is immediately replaced with a spare one. Let $t_k^{(j)}$ be the moment of the *k*th replacement during the *j*th Monte Carlo experiment. The number of spare units of type *i* spent at moment $t_k^{(j)}$ is denoted $x_{ik}^{(j)}$.

An initial state at $t_0^{(j)}$ =0 is:

$x_{i0}^{(j)}$ for all *i*, 1, 2, …4, and *j*, *j*=1, 2, ..., *N*.

The total cost of spare units at the initial moment is C_0 = 0. (Sometimes it might be reasonable to consider the initial cost of the system with no spare units as C_0, that is,

$C_0 = \sum_{1 \le i \le n} k_i c_i$ where k_i is the number of units of type i within equipment before reliability improvement.)

Consider the step by step procedure of generating trajectories $\psi^{(s)}$, s=1, 2, ... We begin with $\psi^{(1)}$ but the corresponding subscript, (1), will be omitted for the sake of convenience.

Step 1. Generate random time to failure (TTF) for each unit, and define $b_{i1}=\xi_i$, that is b_{i1} is the moment of the earliest failure (and instantaneous replacement) of the ith unit. The current moment (for every unit i) at the beginning of any trajectory $\psi^{(s)}$ is $b_{i0}=0$.

Step 2. Determine the moment of the occurrence of the first event (first replacement) within the first realization $\psi^{(1)}$ as $t_1 = \min_{1 \le i \le n} b_{i1}$.5

Step 3. Assign to the corresponding unit (for which the moment of failure is the earliest one) a specific number $i=i_1$.

Step 4. Put into the spare units counter a new value $x_{i_1 1} = x_{i_1 0} + 1$.6

Step 5. Rename remaining x_{i0} as follows: $x_{i0}=x_{i1}$ for all $i \neq i_1$.

Step 6. Calculate a new value of the system cost $C_1=C_0+c_{i_1}$ 7

Step 7. Generate the next random TTF for unit i_1, ξ_{i_1}.8

Step 8. Calculate the next event occurring due to unit i_1: $b_{i_1 2} = t_1 + \xi_{i_1}$.9

Step 9. Rename the remaining values $b_{i1}=b_{i2}$ for all $i \neq i_1$.

His completes the first cycle. GOTO Step 2, i.e. find $t_2 = \min_{1 \le i \le n} b_{i2}$ 10, and so on until stopping the first realization.

The type of problem to be solved determines the stopping rule of each realization.

Stopping rule for the Inverse Problem of optimal redundancy: The process is stopped at the moment t_N when the total cost of spare units exceeds the permitted C^0.

Stopping rule for the Direct Problem of optimal redundancy: The simulation process for each realization stops at the moment $t_{M-1} < t^* \le t_M$ where t^* equals the required operational time t_0 (if the reliability index is the probability of failure free operation) or t^* is the required system's MTTF.

After the termination of generating the first trajectory, $\psi^{(1)}$, we start to generate $\psi^{(2)}$ by the same rules. The number of needed realizations, N, is determined by the required accuracy of statistical estimates.

Thus, each trajectory j represents a set of the following data:

$$\begin{gathered}\{t_1^{(j)}; X_1^{(j)}; C(X_1^{(j)})\}\\ \{t_2^{(j)}; X_2^{(j)}; C(X_2^{(j)})\}\\ \ldots\\ \{t_M^{(j)}; X_M^{(j)}; C(X_M^{(j)})\}\end{gathered}$$

where $X_s^{(j)}$ is the set of spare units at moment $t_s^{(j)}$, i.e., $X_s^{(j)} = \{x_{1s}^{(j)}, x_{2s}^{(j)}, ..., x_{ns}^{(j)}\}$.

After the description of the Monte Carlo simulating process, let us consider the optimization problems themselves. We cn make an important remark: previously all problems were formulated in probabilistic terms, but dealing with statistical (empirical) functions has its specific. Below these problems are reformulated in an appropriate way.

8.4.Description of the Idea of the Solution

Assume that we need to supply some system with spare units for a specified period of time. We have no prior knowledge on units reliability but we have an opportunity to observe a real process (or simulation) of failure occurrence.

Consider the direct problem of optimal redundancy. What shall we do in this case? We observe the process of spare unit expenditure during time *t**. This process can be described as a random travel – call it trajectory – in discrete *n*-dimensional space X. Illustration of such process –n two-dimensional case is presented bellow.

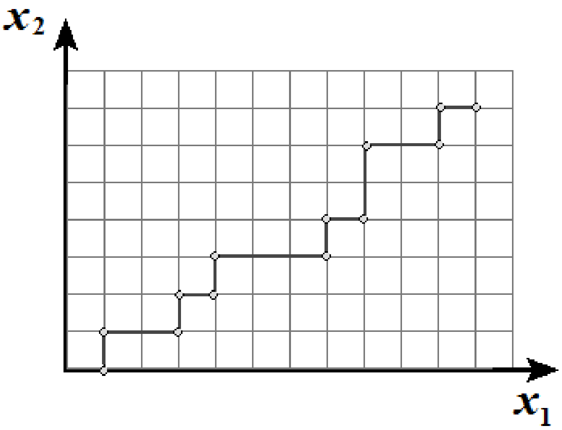


Figure 8.1. Example of a two-dimensional trajectory.

Let us observe *N* such trajectories, *j*=1, 2, ... , *N* in an *n*-dimensional space where *n* is the number of unit types. Each realization is stopped when the total cost of spare units exceeds the permitted amount, that is, each trajectory reaches or even penetrates a hyper plane

$$\Gamma = \{C(X) = C^0\} \qquad (8.3)$$

determined by the restriction on the total system cost (example for two-dimensional case is given in the figure below).

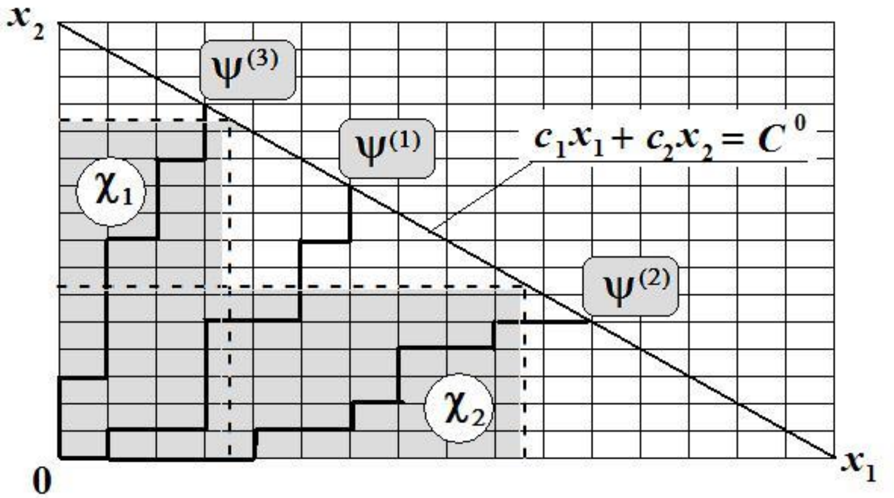


Figure 8.2. Example of two-dimensional trajectory inside hyper plane $C(x_1, x_2)$=C^0.

After this, in the same *n*-dimensional space, we construct such hypercube χ_r, r = 1, 2, ... that each of their vertexes is lying under the hyper plane). In Figure 8.2 there are two such hypercube though there could be many of them: actually in this case all pairs (x_1, x_2) that belong to hyper plane $C(x_1, x_2)$=C^0 could be vertices of such hypercube.

Denote the maximum time that trajectory $\psi^{(j)}$ is spending within the hypercube χ_r by $\tau_r^{(j)} = \tau(\psi^{(j)}, \tau_r)$. Introduce an indicator

$$\delta_r^{(j)} = \begin{cases} 1 \text{ if } \tau_r^{(j)} \geq t_0 \\ \\ 0 \text{ otherwise.} \end{cases} \qquad (8.4)$$

Among all hypercubes above we choose such hypercube χ_r that maximizes the frequency of failure-free operation during required interval t_0 under the cost restrictions:

$$\max_{\chi_r}\left\{\frac{1}{N}\sum_{1\leq j\leq N}\delta_r^{(j)}\left|\sum_{\substack{1\leq i\leq n\\ x_i\in\chi_r}} x_i c_i \leq C^0\right|\right\}. \qquad (8.5)$$

where C^0 is the admissible redundant group cost.

Maximization of the system average time to failure is reached by the hypercube χ'_r that corresponds to the solution of the following problem of the conditional optimization:

$$\max_{\chi'_r}\left\{\frac{1}{N}\sum_{1\le j\le N}\tau_r^{(j)}\left|\sum_{\substack{1\le i\le n\\ x_i\in\chi'_r}} x_i c_i \le C^0\right.\right|. \qquad (8.6)$$

Now consider the Direct Problem of optimal redundancy. In this case, the required time of the system failure-free operation equals t_0. We observe the process of spare units expenditure N times until the system failure-free time exceeds t_0 and record trajectories $\psi^{(j)}$, $j = 1, 2, \ldots, N$ in an n-dimensional space. Afterwards we construct such hypercubes χ_r, $r = 1, 2, \ldots$ in the same n-dimensional space that include (cover) $R^0 \times 100\%$ of all trajectories where R^0 is the specified level of the reliability index. Among all hypercubes described above, we choose the one that is characterized by the minimum total cost. In other words, the hypercube $\tilde{\chi}_r$ must satisfy the solution of the following problem:

$$\min_{\tilde{\chi}_r}\left\{\sum_{\substack{1\le i\le n\\ x_i\in\tilde{\chi}_r}} c_i x_i \left| \frac{1}{N}\sum_{1\le j\le N}\delta_r^{(j)} \ge R^0\right.\right\}. \qquad (8.7)$$

Now let the specified requirement be given for the system average time to failure, T^*. The hypercube $\tilde{\chi}'_r$ presenting the solution, must be chosen corresponding to the solution of the problem:

$$\min_{\tilde{\chi}_r}\left\{\sum_{\substack{1\le i\le n\\ x_i\in\tilde{\chi}_r}} c_i x_i \left| \frac{1}{N}\sum_{1\le j\le N}\tau_r^{(j)} \ge T^0\right.\right\}. \qquad (8.8)$$

Of course, one should take into account that the operation with the frequency differs from the operation with the probability. The proposed solution is asymptotically accurate. Thus all these arguments are satisfactory only for a large enough sample size.

8.5.Inverse Optimization Problem

8.5.1. System Successful Operation vs. System Cost

We need to find χ^{opt} that satisfies (8.5). The algorithm for solution is as follows.

Step 1. Choose a hypercube χ_1 whose diagonal vertex is lying on or under the hyperplane (8.3).

Step 2. Take the first realization, $\psi^{(1)}$ obtained with the help of the Stopping Rule 1. Find moment $\tau_1^{(1)}$ when this trajectory "punctures" the hypercube χ_1. This corresponds to the moment $t_k^{(1)}$ where

$$k = \{k : (\forall x_{i,k-1} \leq \chi_{i1}) \wedge (\exists x_{ik} > \chi_{i1}) \text{ IS TRUE}\}.$$

where χ_{i1} is a component of χ_1 .

Step 3. Assign to $\tau_1^{(1)}$ a value 1 or 0 by using the indicator function (8.4).

Step 4. Add $\delta_1^{(1)}$ to the value in the counter (initial value equals 0) of successful trajectories.

Repeat the procedure from Step 2 for the next realization, $\psi^{(2)}$.

After the analyzing of all trajectories, we calculate the frequency of successful trajectories, $\hat{P}_1(t_0) = \dfrac{1}{N} \sum\limits_{1 \leq j \leq N} \delta_1^{(j)}$ 16 for the chosen hypercube χ_1 . Then we find such hypercube χ_K that is characterized by the maximum value of $\hat{P}_K(t_0)$. For this purpose, we can use a random search, steepest descent, or other numerical optimization procedure in the discrete space of the trajectories of the spare units expenditure.

Numerical example.

Consider a series system of n=3 units. For the sake of simplicity of illustrative calculations and possibility to compare an obtained solution with analytical solution, assume that the system units are independent and $c_i = c$ for all i, $1 \leq i \leq 3$. Let the unit TTF be distributed exponentially with parameters $\lambda_1 = 1$, $\lambda_2 = 0.5$, and $\lambda_3 = 0.25$, respectively. The specified time of failure-free operation is t_0=1. Admissible total cost of a spare unit is equal to 4.

In the left three columns of Table 8.1, there are random exponentially distributed time-to-failure, ξ_i , for all three units. In the next three columns there are corresponding sequences of replacement times $\theta_{i(k)}$: $\theta_{i(k)} = \xi_{i1} + \xi_{i2} + ... + \xi_{ik}$ are also there in the same table. In other words, $\theta_{i(k)}$ is a random survival time of the ith redundant group consisting of one main and k-1 spare units.

Table 8.1. Random TTF and replacement time for 10 Monte Carlo realizations

TTF			Replacement time		
unit 1	unit 2	unit 3	unit 1	unit 2	unit 3
Realization 1					
0.07	0.75	0.24	0.07	0.75	0.24
0.53	4.97	3.19	0.6	5.72	3.43
0.06	0.45	1.41	0.66	6.17	4.84
0.53	2.59	3.42	1.19	8.76	8.26
1.44	5	1.59	2.63	13.76	9.85
Realization 2					
0.42	0.13	4.92	0.42	0.13	4.92
0.16	1.15	12.9	0.58	1.28	17.82

0.45	3.29	0.83	1.03	4.57	18.65
0.28	0.35	2.35	1.31	4.91	21
0.25	2.1	1.74	1.55	7.02	22.73
Realization 3					
0.62	3.47	3.22	0.62	3.47	3.22
3.66	2.72	3.92	4.28	6.2	7.14
5.11	2.47	3.9	9.39	8.67	11.04
0.31	1.69	1.21	9.7	10.36	12.25
1.42	0.86	0.96	11.12	11.22	13.21
Realization 4					
1.45	5.85	0.51	1.45	5.85	0.51
1.13	1.26	8.64	2.58	7.11	9.15
1.27	2.14	4.71	3.85	9.25	13.86
0.45	0.67	1.16	4.29	9.92	15.01
2.48	1.52	6.38	6.77	11.44	21.4
Realization 5					
0.32	0.22	0.54	0.32	0.22	0.54
0.75	0.15	1.53	1.08	0.37	2.06
0.73	1.49	1.78	1.81	1.87	3.84
0.01	0.68	0.89	1.82	2.55	4.73
0.25	3.06	1.68	2.07	5.6	6.41
Realization 6					
0.11	2.03	5.54	0.11	2.03	5.54
1.03	0.48	10.57	1.13	2.52	16.11
0.88	2.26	5.14	2.01	4.77	21.25
0.39	5.19	0.92	2.41	9.96	22.17
3.45	1.12	6.58	5.86	11.08	28.74
Realization 7					
1.22	0.11	2.69	1.22	0.11	2.69
1.87	0.91	0.1	3.09	1.02	2.79
0.41	2.11	1.9	3.5	3.13	4.69
3.95	0.36	3.72	7.45	3.49	8.41
0.4	1.67	1.43	7.85	5.17	9.84
Realization 8					
0.27	1.49	22.49	0.27	1.49	22.49
0.44	0.53	1.24	0.71	2.02	23.73
0.74	1.07	12.07	1.45	3.09	35.8
0.76	1.13	2.86	2.2	4.22	38.65
0.36	2.99	2.87	2.57	7.21	41.52
Realization 9					
0.46	1.55	7.9	0.46	1.55	7.9
1.06	4.8	7.59	1.52	6.35	15.49
1.9	2.66	8.14	3.42	9.01	23.63
0.17	0.37	1.26	3.59	9.38	24.89
2.18	0.43	5.17	5.77	9.8	30.06

Realization 10					
0.83	1.08	0.58	0.83	1.08	0.58
0.4	1.76	3.76	1.23	2.84	4.33
1	0.94	8.73	2.23	3.79	13.07
0.4	1.48	3.74	2.63	5.26	16.81
0.47	2.91	3.73	3.1	8.18	20.54

Remark: We will use the same random numbers for all of the examples below. This leads to a dependence of the results obtained for different problems, but our main goal is to illustrate the algorithm of the solution with the use of a numerical example, rather than to execute an accurate statistical experiment.

Numerical solution

(1) Consider Realization 1 from Table 8.1. First, take values ξ_i of the first row of the left block of columns: ξ_{11}= 0.07, ξ_{21}= 0.75 and ξ_{31}= 0.24. Denote them $\theta_{1(1)}$, $\theta_{2(1)}$ and $\theta_{3(1)}$, respectively, and set them into the first row of the right block of columns ("Replacement time"). Find the minimum value: min {$\theta_{1(1)}$, $\theta_{2(1)}$, $\theta_{3(1)}$}=$\theta_{1(1)}$ =0.07.

(2) Next take the value ξ_{12}= 0.53 in the column "TTF; Unit 1". Form a new value: $\theta_{1(2)}$ = $\theta_{1(1)}+\xi_{12}$ = 0.07 + 0.53 = 0.6. Rename $\theta_{2(1)}$ = $\theta_{2(2)}$ and $\theta_{3(1)}$ = $\theta_{3(2)}$. Set this value into the second place in the column "Replacement time; Unit 1".

(3) Find the minimum value: min {$\theta_{1(2)}$, $\theta_{2(2)}$, $\theta_{3(2)}$}=$\theta_{3(2)}$ = 0.24. Repeat step (2) until the total cost of each system equals 7. (For the case c_i=c, it means that all 7 units are spent.) As the result, we spent three units of type 1, no units of type 2, and two units of type 3. In this particular case, the system TTF does not reach the specified time t_0=1.

(4) Repeat steps (1) to (3) with the remaining realizations from Table 8.1 and fill out Table 8.2.

Table 8.2. Initial experiment with exclusion of "extra units" (marked with "*")

Unit-2	**Unit-2**	**Unit-3**
Realization 1		
0.07	0.75	0.24
0.6		3.43
0.66		
1.19		
Realization 2		
0.42	0.13	4.92
0.58	1.28	
1.03		
1.31*		
Realization 3		
0.62	3.47	3.22
4.28	6.2*	7.14*

9.39*		
Realization 4		
1.45	5.85	0.51
2.58*	9.15	
3.85*		
4.29*		
Realization 5		
0.32	0.22	0.54
1.08	0.37	2.06
	1.87	
Realization 6		
0.11	2.03	5.54
1.13	2.52*	
2.01*		
2.41*		
Realization 7		
1.22	0.11	2.69
3.09*	1.02	2.79*
	3.13*	
Realization 8		
0.27	1.49	22.49
0.71	2.02*	
1.45		
2.2*		
Realization 9		
0.46	1.55	7.9
1.52	6.35*	
3.42*		
3.59*		
Realization 10		
0.83	1.08	0.58
1.23	2.84*	4.33
2.23*		

(5) Notice that units marked with "*" in Table 8.2 are auxiliary, that is, in each particular case they are not necessary because t_0 has been reached before all permitted resources were spent.

(6) In Table 3, list all vectors: $X^{(k)} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)})$, k=1,2, ... ,10, that are obtained from Table 8.2 after exclusion of the marked units (see Table 8.3).

Table 8.3. Realization of units spent (corrected for $t_0 \geq 1$)

Realization Number	x_1	x_2	x_3	System's TTF
#1	4*	1*	2*	<1

#2	3	2	1	1.03
#3	2	1	1	3.22
#4	1	1	2	1.45
#5	2	3	2	1.08
#6	2	1	1	1.13
#7	1	2	1	1.02
#8	3	1	1	1.45
#9	2	1	1	1.55
#10	2	1	2	1.08
Maximum	**3**	**3**	**2**	

Remark: Realization #1 is not taken into account because its TTF<1.

(7) Order each component of these vectors separately (see Table 8.4). In other words, Table 8.4 shows the frequency with which a corresponding number of spare units of each type has been met during 10 realizations of the Monte Carlo simulation. We see that the use of the vector (3, 3, 2) of spare units for this realization will lead to 1 failure in 10 experiments. However, the total system cost equals 8 units. So, the next step is to find the best way of reducing the total system cost.

Table 8.4. Ordered numbers of the use of units of different types

x_1	x_2	x_3
1	1	1
2	1	1
2	1	1
2	1	1
2	1	1
2	(1)	2
3	2	2
3	2	2
3	2	2
(4)	3	(2)

Note: Numbers in parenthesis correspond to the first realization which was not taken into account.

(8) Put the number of realizations for which TTF has not reached t_0=1 into the failure counter. In our case there is only one such realization with TTF ≤ 1.
(9) Exclude from Table 8.2 all vectors which correspond to the realizations mentioned in step (7).
(10) Find which unit in Table 8.4 has the smallest number of the use of largest values of $\max\limits_{1\leq k\leq 10} x_i^{(k)}$. In our example, three units of type 1 were used in three realizations, three units of type 1 were used once, and two units of type 3 were used four times. (We exclude from consideration Realization #1 since it did not deliver TTF$\geq$1.)

In this case we exclude one unit of type 2 because in this case we gain one unit of cost and "loss" only one realization.

(11) Add number of units excluded at the step (9) into the counter of system failures.
(12) Check if the system cost equal to or less than C^0 =7. If "No", correct Table **11**.3 by exclusion of the vector 1 and continue the procedure from step (6). If "Yes", stop the procedure and
(13) calculate the ratio of realization without failure (the total number of realization minus the number of failures from the counter) to the total number of performed realizations.

In the example considered the final solution is (3, 2, 2).
As a direct calculation with the use of tables of Poisson distribution shows, this vector delivers the probability 0.804. Of course, such a coincidence with the observed frequency equal to 0.8 in a particular statistical experiment is not a proof of the method. However, multiple results obtained by the proposed method for different other examples show a proper closeness to the exact solution even for a relatively small sample size.

The asymptotical convergence of the solution to the optimal one was proved in [Gordienko & Ushakov, 1978].

8.5.2. System Average Time to Failure vs. System Cost

We need to find χ'_{opt} that satisfies the solution of (8.6). In this case the algorithm almost completely coincides with the one described above. The only difference is in the absence of Step 3. At Step 4 we put directly $\tau_1^{(i)}$ in a counter of the survival time. After analyzing all of the trajectories, the estimate of the MTTF for the hypercube χ_1 is calculated as

$$T_1 = \frac{1}{N} \sum_{1 \le i \le N} \tau_1^{(1)}. \qquad (8.9)$$

After this we perform the analogous calculations for other hypercubes finding such of them that are characterized by the maximum estimate of MTTF. The search for the maximum can be performed in the same way as was done previously.

Example 8.5. We will consider the same data as in the example above. The system is again allowed to have at most 7 units in total.
Numerical solution

Repeat steps from 1 to 4 of that described in Section 3. In other words, we assume that Table 8.2 is constructed. For solution of this problem, we will use all data of Table 8**.2.** The continuation of the algorithm for this case is as follows.

(5) Consider vectors of Table 8.2. In this case, the components marked with "*" are included. Those vectors are obtained in the imitation process until 7 units of price are spent. Now extract corresponding values from the right side of Table 8.1 (see Table 8.5).

Table 8.5. Random time to failure for each realization until expenditure of seven units.

unit 1	unit 2	unit 3
Realization 1		
0.07	0.75	0.24
0.53		3.19
0.06		
0.53		
Realization 2		
0.42	0.13	4.92
0.16	1.15	
0.45		
0.28		
Realization 3		
0.62	3.47	3.22
3.66	2.72	3.92
5.11		
3.9		
Realization 4		
1.45	5.85	0.51
1.13		8.64
1.27		
0.45		
Realization 5		
0.32	0.22	0.54
0.75	0.15	1.53
	1.49	
Realization 6		
0.11	2.03	5.54
1.03	0.48	
0.88		
0.39		
Realization 7		
1.22	0.11	2.69
1.87	0.91	0.1
	2.11	
Realization 8		
0.27	1.49	22.49
0.44	0.53	
0.74		
0.76		
Realization 9		
0.46	1.55	7.9
1.06	4.8	
1.9		
0.17		

Realization 10		
0.83	1.08	0.58
0.4	1.76	3.76
1		

In this table we can see for how long each unit was operating.

(6) On the basis of this table, we compose Table 8.6. In each position of this table we have the total sum of the time spent during all realizations. First of all, for independent and identical units, these values depend on the number of realizations where this unit was observed. (In general case, where units are different and could be dependent, the number of such realizations might not be a dominant parameter.) These values from the bottom show how much we will loose by excluding a unit.

Table 8.6. Sum of the times spent by units on the specified positions

unit 1	unit 2	unit 3
5.77	16.68	48.63
11.03	12.5	21.14
11.41	3.6*	3.9*
2.58*		

"*" denotes the units which are eliminated (x_1 = 4, x_2 = 3, and x_3 = 3).

It is clear that the loss will be less if we leave x_1 = 3, x_2 = 2 and x_3 = 2. By eliminating them we decrease the total system cost up to 7 units of price.

The time to failure for the system in each realization is calculated as the minimal value among those, which are restricted by vector (x_1=3, x_2=2,x_3), that is, for the *k*th realization

$$\xi^{(k)} = \min\{\xi_{13}^{(k)}, \xi_{22}^{(k)}, \xi_{32}^{(k)}\} \ \xi_{Syst}^{(k)} \quad = \quad \min(\xi_{13}^{(k)},\ \xi_{22}^{(k)},\ \xi_{32}^{(k)})$$

These values can be found on the right side of Table 8.6. The results are shown in Table 8.7. These values allow calculating the mean time to failure of the investigated system.

Table 8.7. Time to failure for 10 realizations picked up for vector (3, 2, 2)

Realization number	TTF
1	0.66
2	1.03
3	6.2
4	3.85
5	0.37
6	2.01
7	1.02

8	1.45
9	3.42
10	2.23

8.6. Direct Optimization Problem

8.6.1. System Cost vs. Successful Operation

We need to find $\tilde{\chi}_r$ which satisfies the solution of (8.5). The algorithm of solution in this case is as follows.

Step 1. Construct a realization of the first trajectory of the spare unit expenditure until $t_1^{(1)}$ exceeds the specified value of operational time t_0. Memorize the number of spare units spent, $x_i^{(1)}$, i = 1, 2, ..., n. Continue this procedure until all of N required trajectories are constructed.

Step 2. Construct a hypercube χ_1 whose edges χ_{i1} are found as $\chi_{i1} = \max\limits_{1 \le j \le N} x_{ij}$, that is, χ_{i1} is the maximum number of spare units of type i observed during all N realizations. (It means that for this particular sample of realizations, all of them will lay within such a hypercube that is with such a stock of spare units we would not observe any system failure.)

Step 3. Calculate the system cost for the hypercube $\chi^{(1)}$ for which all trajectories have the survival time no less than t_0

$$C_{\max} = \sum_{1 \le i \le n} c_i \;\; \chi_{i1}.$$

Step 4. Calculate for each i:

$$\gamma_i^{(1)} = \frac{\nu_i^{(1)}}{\Delta c_i^{(1)}}$$

where $\nu_i^{(1)}$ shows how many numbers equal to χ_{i1} exist for a unit of type i and $\Delta c_i^{(1)}$ is the value of the system cost decrease if we reject to use $\max\limits_{1 \le j \le N} \chi_{ij}$ and will use the next value in the descending order.

Step 5. Find the type of units which correspond to the maximum value of $\gamma_i^{(1)}$ and name it as i_1, that is, this number corresponds to the following condition:

$$i_1 = \left\{ i : \gamma_i = \max_{1 \le j \le n} \gamma_j^{(1)} \right\}.$$

Step 6. Exclude $\nu_{i_1}^{(1)}$ units of type i_1 and form a new value

$$\chi_{i_1 2} = \chi_{i_1 1} - \nu_{i_1}^{(1)}.$$

Step 7. Rename remaining numbers

$$\chi_{i_1 2} = \chi_{i_1 1} \text{ for all } i_j \neq i_1$$

Step 8. Calculate the system successful operation index after the exclusion of ν_{i_1} units of type i_1

$$\hat{P}^{(2)} = 1 - \frac{\nu_{i_1}^{(1)}}{N}$$

Step 9. Calculate the system spare units cost after the exclusion of ν_{i_1} units of type i_1

$$C^{(2)} = C_{\max} - c_{i_1} \nu_{i_1}^{(1)}.$$

After these steps we have a new hypercube χ_2 :

$$\chi_2 = \{\chi_{12}, \chi_{22}, \dots, \chi_{n2}\}.$$

Repeat the procedure from step 5 until the system spare units cost is equal to or smaller than the given restriction.

<u>**Numerical example**</u>.

Let us take $\hat{R}(X) \ge 0.9$. In the previous example we found that the vector of spare units (4, 3, 2) satisfies 100% of successful realizations. So, if we take a vector (3, 3, 2) it will satisfy the condition $\hat{R}^0 = 0.9$. Now we need to find the lower 90% confidence limit for the frequency 0.9 obtained in 10 experiments. This limit can be found with the use of Clopper-Pearson method.

In this particular case it is easier to make direct calculations. If we choose the estimate of the searched probability equal to 0.9 then the probability that we will observe no less than 8 successes is equal to

$$0.9^{10} + \binom{10}{1}(0.1)(0.9)^{9} + \binom{10}{2}(0.1)^{2}(0.9)^{8} = 0.9298.$$

So, in the process of decreasing the number of used spare units, we must stop after the first exclusion, i.e., the solution in this case is (3, 3, 2).

Thus, the solutions of direct and inverse problems of optimal redundancy are different though they should coincide. The difference lies in the difference of approaches: having the restriction on the system cost, we maximize the possible observed *frequency*; in the latter case we consider minimization of the system cost under condition that the level of *probability* is guaranteed. This difference will be smaller if the number of realizations is larger.

Remark: We could solve the problem above with an iteration procedure using the solution of the direct problem of optimal redundancy. The use of the "fork method" is convenient in this case. We find the solution, χ_1^{opt} for some cost restriction, say, C_1^*, and calculate the value $R_1 = R(\chi_1^{opt})$. If $R_1 < R^*$ we chose $C_2^* > C_1^*$ and continue the procedure; if $R_1 > R^*$ we chose $C_2^* < C_1^*$ and also continue the procedure. For the next steps, we can use a simple linear approximation

$$C^*_{k+1} \ = \ C^*_k \ - \ (C^*_k - C^*_{k-1}) \frac{R_k - R^*}{R_k - R_{k-1}}.$$

where subscript k stands for the current step, subscript $k - 1$ for the previous step, and subscript $k + 1$ for the next step.

8.6.2.System Cost vs. Average Time to Failure

We need to find $\tilde{\chi}_{r'}$ which satisfies the solution of (8.6). We could not find a convenient procedure for solving this particular problem. One might consider using an iteractive procedure using the sequential solution of the second direct problem considered above. For instance, we can fix some restriction on the system cost, say, $C_{syst}^{(1)}$ and find the corresponding optimal solution for $\hat{T}_{syst}^{(1)}$. If this value is smaller than the required $\hat{T}_{syst}^{*}$, it means that the system cost must be increased, say, up to some $C_{syst}^{(2)} > C_{syst}^{(1)}$. If $\hat{T}_{syst}^{(1)} > \hat{T}_{syst}^{*}$, one must choose $C_{syst}^{(2)} < C_{syst}^{(1)}$. This procedure continues until a satisfactory solution is obtained. At an intermediate step L for choosing $\hat{T}_{syst}^{(L)}$, one can use some linear extrapolation method. For example, assume that in first situation described above, the value $\hat{T}_{syst}^{(2)}$ is still less than $\hat{T}_{syst}^{*}$. Then the value of $C_{syst}^{(3)}$ can be chosen from the following equation

$$\frac{C_{syst}^{(3)} - C_{syst}^{(1)}}{C_{syst}^{(2)} - C_{syst}^{(1)}} = \frac{T_{syst}^{*} - T_{syst}^{(1)}}{T_{syst}^{(2)} - T_{syst}^{(1)}}.$$

Obviously, one can also use the procedure similar to that in a solution of the direct problem. However one should somehow find an initial hypercube and construct all trajectories within it. (There is no stopping rule in this case.) Then one should construct a system of embedded hypercubes and again use the steepest descent.

While solving this problem one must remember that the condition $\hat{T}_{syst} \geq T^{*}$ can be considered only in a probabilistic sense.

Bibliography to Chapter 8

1977. Ushakov, I.A., and Yasenovets , A.V. Statistical methods of solving problems of optimal standby. *Soviet Journal of Computer and System Sciences*, No. 6,.

1978. Ushakov, I. A., and Gordienko, E.I. On statistical simulation approach to solution of some optimization problems. Elektronische Informationsverarbeitung und Kybernetik, No.3.

1978. Ushakov, I.A., and Gordienko, E.I. Solution of some optimization problems by means of statistical simulation. *Electronosche Infdormationsverarbeitung und Kybernetik*, No.11.

1988. Mohan, C., and Shanker, K. Reliability optimization of complex systems using random search technique. Microelectronics and Reliability, No. 28.

1992. Boland, P.J., El-Neweihi, E., and Proschan, F. Stochastic order for redundancy allocations in series and parallel systems. Advances in Applied Probability, No. 1.

2003. Zhao, R., and Liu, B. Stochastic programming models for general redundancy-optimization problems. IEEE Transaction on Reliability, No. 52.

9.COMMENTS ON CALCULATION METHODS

9.1. Comparison of methods

Optimal redundancy is a very important practical problem. The solution of the problem allows one to improve the reliability at a minimal expense. But here, as in many other practical problems, questions arise: What is the confidence of the obtained results? What is the real effect of the use of sophisticated mathematics?

These are not unreasonable questions!

We already have discussed what it means to design an "accurate" mathematical model. It is always better to speak about a mathematical model which more or less correctly reflects a real object. But let us suppose that we “almost sure” that the model is perfect. What price are we willing to pay for obtaining numerical results? What method is best, and best in what sense?

The use of excessively accurate methods is, for practical purposes, usually not necessary because of the uncertainty of the statistical data. On the other hand, it is inexcusable to use approximate methods without reason.

We compare the different methods in the sense of their accuracy and computation complication.

The Lagrange Multiplier Method (LMM) demands the availability of continuous, differentiable functions. This requirement is met very rarely: one usually deals with essentially discrete nature of the resources. But LMM sometimes can be used for a rough estimation of the desired solution.

The Steepest Descent Method (SDM) is very convenient from a computational viewpoint. It is reasonable to use this method if the resources that one might spend on redundancy are large. Of course, this generally coincides with the requirement of high system reliability because this usually involves large expenditures of resources.

But unfortunately, it happens very rarely in practice. At any rate, one can use this approach for solution of most practical problems without hesitation.

The absolute difference between costs of the two neighboring SDM solutions cannot exceed the cost of the most expensive unit value. Thus, it is clear that the larger the total cost of the system, the smaller the relative error of the solution.

The Dynamic Programming Method (DPM) and its modifications (Kettelle's Algorithm and the Method of Universal Generating Function) are exact but they demand more calculation time for and a larger computer memory. As with most discrete problems requiring an enumerating algorithm, these optimal redundancy problems are *np*-hard.

As we mentioned above, the SDM may provide even an absolutely exact solution, since a dominating sequence for SDM is a subset of dominating sequence of DPM.

In figures below, one finds two solutions obtained by SDM and DPM .

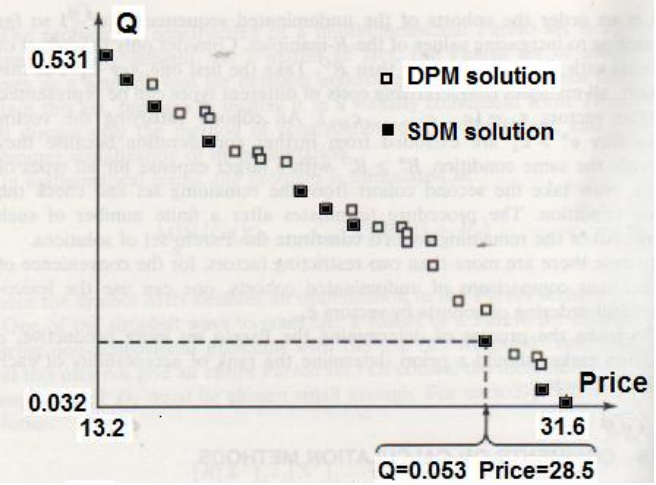


Figure 9.1. Comparison DPM nad SDN solutions.

The left figure contains dominating sequence obtained by DPM, and the right one does o those obtained by SDM. The dots of the left figure marked with black color corresponds to the dots of the right figure.

Of course, one of the questions of interest is the stability of the solutions. How does the solution depend on the accuracy of the input data? How does the solutions obtained by the use of different methods distinguished? How much the numerical results of the solutions differ from one method to another?

An illustration of the problem is given by numerical experiments.

Numerical example.

Consider a series system consisting of three units. The input data are assumed to be uncertain: units' PFFO and cost are known with an accuracy of 10%. To demonstrate possible difference in solutions, let us take the following five systems:

Table 9.1.

	Unit-1		Unit-2		Unit-3	
System	*q*	*c*	*q*	*c*	*q*	*c*
A	0.2	1	0.2	1	0.2	1
B	0.2	0.9	0.2	1	0.2	1.1
C	0.18	0.9	0.2	1	0.22	1.1
D	0.18	1.1	0.2	0.1	0.22	0.9
E	0.18	1	0.2	1	0.22	1

The problem is to check the stability of the optimal solutions over the range of variability of the parameters.

Solution

At first, we compare the solutions for all five cases if the specified total system cost is to be at most 30 units. For each case we give two results: one obtained by the SDM and the second (marked with *) obtained by the DPM. The results are as follows:

Table 9.2.

	Number of redundant units			Probab. of	Factual
System	x_1	x_2	x_3	syst. failure	syst. cost
A	10	10	10	$3.07 \cdot 10^{-7}$	30
A*	10	10	10	$3.07 \cdot 10^{-7}$	30
B	10	10	10	$3.07 \cdot 10^{-7}$	30
B*	10	10	10	$3.07 \cdot 10^{-7}$	30
C	9	10	10	$5.66 \cdot 10^{-7}$	29.1
C*	10	10	10	$4.04 \cdot 10^{-7}$	30
D	9	10	11	$3.59 \cdot 10^{-7}$	29.8
D*	9	10	11	$3.59 \cdot 10^{-7}$	29.8
E	9	10	11	$3.59 \cdot 10^{-7}$	29.8
E*	9	10	11	$3.59 \cdot 10^{-7}$	29.8

The table shows that the only differences between the approximate and exact solutions are observed for the cases C and C*. However, all solutions are very close.

With an increase in spent resources, the relative difference between the solutions obtained by the SDM and the DPM will be increasingly smaller.

We now analyze the solutions corresponding to a specified level of reliability. In the table below for the same systems respective results for $Q_0=1\cdot 10^{-6}$ are collected.

Table 9.3.

System	Number of redundant units			Probab. of syst. failure	Factual syst. cost
	x_1	x_2	x_3		
A	9	10	10	$7.7\cdot 10^{-7}$	30
	Equivalent solution are (10.9,10) and (10,10,9)				
A*	9	10	10	$7.17\cdot 10^{-7}$	30
B	10	10	9	$7.17\cdot 10^{-7}$	30
B*	10	10	9	$7.17\cdot 10^{-7}$	30
C	9	9	10	$9.76\cdot 10^{-7}$	29.1
C*	9	9	10	$9.76\cdot 10^{-7}$	30
D	9	9	10	$9.76\cdot 10^{-7}$	29.8
D*	9	9	10	$9.76\cdot 10^{-7}$	29.8
E	9	9	10	$9.76\cdot 10^{-7}$	29.8
E*	9	9	10	$9.76\cdot 10^{-7}$	29.8

Numerical computer experiments and practical experience in finally solution of the optimal redundancy problem could develop a keen engineering intuition in the approximate solving of such problems and their sensitivity analysis.

9.2.Sensitivity analysis of optimal redundancy solutions

Solving practical optimal redundancy problems, one can ponder: what is the sense of optimizing if input data are plucked from the air? Indeed, statistical data are so unreliable (especially, in reliability problems ☺) that such doubts have a very good ground.

Not found any sources after searching the answer for this question, the author decided to make some investigation of optimal solutions sensitivity under influence of data scattering.

A simple series system of six units has been considered (see Figure 1). For reliability increase, one uses a loaded redundancy, i.e. if a redundant group k has x_k redundant units, its reliability is

$$P_k(x_k)=1-(1-p_k)^{x_k+1}$$

where p_k is a probability of failure free operation (PFFO) of a single unit k. The total cost of x_k redundant units is equal to $c_k\cdot x_k$, where c_k is the cost of a single unit of type k.

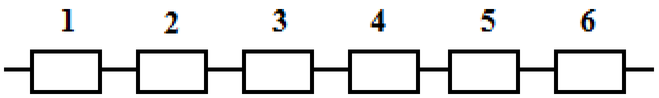


Figure 9.2. Series system underwent analysis.

Units' parameters are presented in Table 9.4.

Table 9.4. Input data.

	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5	Unit 6
p_k	0.8	0.8	0.8	0.9	0.9	0.9
c_k	5	5	5	1	1	1

Assumed that units are mutually independent, i.e. system's reliability is defined as

$$P_{System}(x_k, 1 \le k \le 6) = \prod_{1 \le k \le 6} P_k(x_k)$$

And the total system cost is:

$$C_{System}(x_k, 1 \le k \le 6) = \sum_{1 \le k \le 6} c_k x_k$$

Below solutions of both problems of optimal redundancy are presented: direct:

$$\min_{1 \le x_k < \infty} \left\{ C(x_k, 1 \le k \le 6) \mid P(x_k, 1 \le k \le 6) \ge P^* \right\}$$

and inverse:

$$\max_{1 \le x_k < \infty} \left\{ P(x_k, 1 \le k \le 6) \mid C(x_k, 1 \le k \le 6) \le C^* \right\}$$

For finding the optimal solutions, the Steepest Descent Method was applied. For this "base" system the solutions for several sets of parameters are presented for Direct Problem in Table 9.5 and for Inverse Problem in Table 9.6. (Numbers are given with high accuracy only for demonstration purposes; in practice, one has to use only significant positions after a row of nines.)

Table 9.5. Solution for Direct problem.

P^*	x_1	x_2	x_3	x_4	x_5	x_6	Achieved P	System C
0.95	3	3	3	3	2	2	0.9559520	52
0.99	4	4	3	3	3	3	0.991187	69
0.995	5	4	4	4	3	3	0.995229	75
0.999	6	5	5	4	4	4	0.999218	93

Table 9.6. Solution for Inverse problem.

C^*	x_1	x_2	x_3	x_4	x_5	x_6	Achieved C	System P
50	3	3	2	2	2	2	46	0.931676
75	4	4	3	3	3	3	75	0.995229
100	5	4	4	4	3	3	99.5	0.999602

The questions of interest are: how optimal solution will change if input data are changed? Two types of experiments have been performed: in the first series of experiments, different unit's costs with fixed probabilities were considered (see Figure 9.3) and in another one different unit's probabilities with fixed costs were considered (see Figure 9.4).

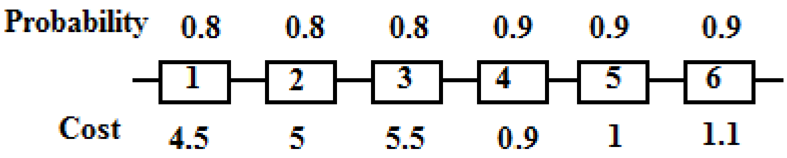


Figure 9.3. Input data for the first series of experiments.

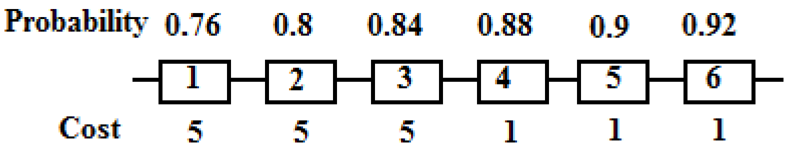


Figure 9.4. Input data for the second series of experiments.

The results of calculations are as follows:

Table 9.7. Values of Probabilities of Failure-free operations.

	0.999	0.995	0.99	0.95
Initial	0.999218	0.99566	0.9922	0.955952
Various *C*	0.998996	0.99566	0.9922	0.955952
Various *P*	0.999218	0.99566	0.9922	0.955952

In addition, a Monte Carlo simulation was performed where parameters of the PFFO and cost were changed simultaneously. In this case, parameters of each unit were calculated (in Excel) as:

p_k=0.8p_k+0.4p_k*RAND()

and

c_k= 0.8c_k+0.4*RAND(),

i.e. considered a random variation of the values within ±20% limits.

The final results for this case are presented in Tables 9.8 – 9.11.

Table 9.8. Results of Monte Carlo simulations for P*=0.999.

No.	**P* = 0.999**							
	P	**C**	x_1	x_2	x_3	x_4	x_5	x_6
1	0.999352	100	6	6	6	4	4	4
2	0.999218	102	6	6	6	5	4	4
3	0.999313	102	6	6	6	4	4	4
4	0.999212	97	5	6	6	4	4	4
5	0.999182	102	6	6	6	4	4	4
6	0.999171	97	6	6	5	4	4	4
7	0.999171	103	6	6	6	4	5	4
8	0.999596	100	6	6	6	4	4	4
9	0.999526	100	6	6	6	4	4	4

10	0.999399	100	6	6	6	4	4	4

Table 9.9. Results of Monte Carlo simulations for P*=0.995.

No.	**P* = 0.995**							
	P	**C**	x_1	x_2	x_3	x_4	x_5	x_6
1	0.995478	84	5	5	5	3	3	3
2	0.996755	85	5	4	4	4	3	3
3	0.995026	85	5	4	5	4	3	3
4	0.996777	79	4	5	5	3	3	3
5	0.996777	84	5	5	5	3	3	3
6	0.995525	79	5	5	4	3	3	3
7	0.996732	85	5	5	5	3	4	3
8	0.996732	85	5	5	5	3	4	3
9	0.995645	84	5	5	5	3	3	3
10	0.99567	84	5	5	5	3	3	3

Table 9.10. Results of Monte Carlo simulations for P*=0.99.

No.	**P* = 0.99**							
	P	**C**	x_1	x_2	x_3	x_4	x_5	x_6
1	0.990147	69	4	4	4	3	3	3
2	0.990965	70	4	4	4	4	3	3
3	0.990229	70	4	4	4	4	3	3
4	0.99185	69	4	4	4	3	3	3
5	0.990389	71	4	4	4	4	4	3
6	0.99107	69	4	4	4	3	3	3
7	0.992185	74	5	4	4	3	3	3
8	0.990422	71	4	4	4	3	4	3
9	0.990893	71	5	4	4	3	3	3
10	0.990466	69	4	4	4	3	3	3

Table 9.11. Results of Monte Carlo simulations for P*=0.95.

No.	**P* = 0.95**							
	P	**C**	x_1	x_2	x_3	x_4	x_5	x_6
1	0.950045	52	3	3	3	3	2	2
2	0.955842	52	3	3	3	3	2	2
3	0.951936	52	3	3	3	3	2	2
4	0.951711	54	3	3	3	2	2	2
5	0.957883	50	3	3	3	3	3	2
6	0.951908	51	3	3	3	2	2	2
7	0.962227	51	3	3	3	2	2	2
8	0.962227	51	3	3	3	3	3	2
9	0.95261	50	3	3	3	3	2	3
10	0.950393	52	3	3	3	3	2	2

Analysis of data presented in Tables 9.8 - 9.11 shows relatively significant difference in numerical results (see Figure 9.5).

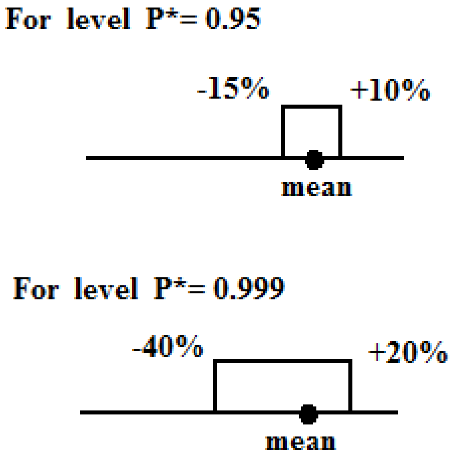


Figure 9.5. Deviation of maximum and minimum values of probability of failure-free operation obtained by Monte Carlo simulation.

However, the problem is not in coincidence of final values of PFFO or cost. The problem is: how the change of parameters influences the optimal values of x_1, x_2, … .

However, one can observe that even with a system of six units (redundant groups) a visual analysis of sets ($\boldsymbol{x_1, x_2, \ldots, x_6}$) is extremely difficult and, at the same time, deductions based on some averages or deviations of various x_k are almost useless.

The author was forced to invent some kind of a special presentation of sets of x_k's. Since there is no official name for such kind of graphical presentation, it is called "multiple polygons". On such multiple polygon there are numbers of "rays" corresponding to the number of redundant of units (groups). Each ray has several levels corresponding to the number of calculated redundant units for considered case (see Figure 9.6).

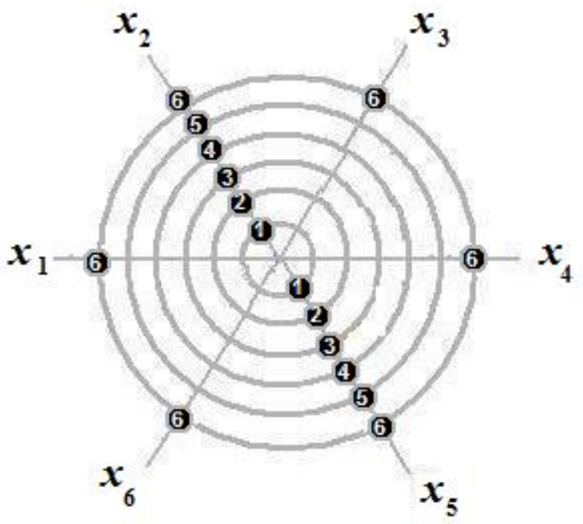


Figure 9.6. Multiple polygon axes with numbered levels.

The multiple polygons give a perfect visualization of "close-to-optimal" solutions and characterize observed deviation of particular solutions. Such multiple polygons for considered example are given in Figure 9.7. (Here bold lines re used for connecting the values of x_k obtained as optimal solution for units with parameters given in Table 9.4.)

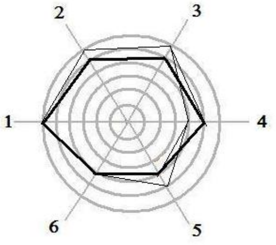
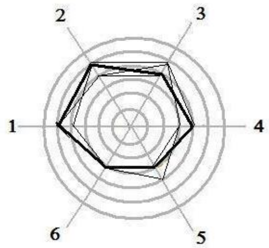
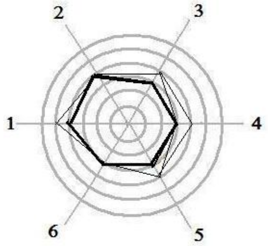
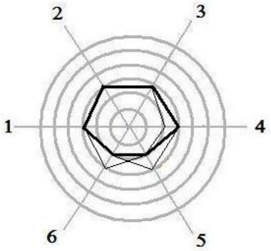
***P** = 0.999**	***P** = 0.999**	***P** = 0.999**	***P** = 0.999**
			

Figure 9.7. Deviations of optimal solutions for randomly varied of parameters from the optimal solution obtained for parameters given in Table 1.

Thus, one can notice that input parameters variation may influence significantly enough the probability of failure-free operation and the total system cost from run to run of Monte Carlo simulation though the optimal solution remains more or less stable.

10.OPTIMAL REDUNDANCY WITH SEVERAL LIMITING FACTORS

10.1. Method of "weighing costs"

A number of cases arise when one has to take into account several restrictions in solving the optimal redundancy problem. For example, various objects such as aircraft, satellites, submarines, etc. have restrictions on cost and also on weight, volume, required electric power, etc. (Apparently, the cost for most of these technical objects is an important factor, but, perhaps, less important than other mentioned.)

In these cases, one has to solve the optimization problem under several restrictions: to maximize the system reliability index, under restrictions on all other factors.

Consider a system consisting of n redundant groups connected in series. For each additional redundant unit of the system, one has to spent some quantity of M various types of resources (for instance, cost, weight, volume, etc.), say, $C_j(X)$. There are constrains on each type of resources: $C_j(X) \le C_j^0$. The optimization problem is formulated as

$$\max_X \left\{ R(X) \,\middle|\, C_1(X) \le C_1^0,\ C_2(X) \le C_2^0,\ \dots, C_M(X) \le C_M^0 \right\} \tag{10.1}$$

where $X = (x_1,\ x_2,\ \dots,\ x_n)$ is the vector of the system redundant units.

Let us assume that each $C_j(X)$ is a linear function of the form

$$C_j(X) = \sum_{1 \le i \le M} c_{ji} x_i \tag{10.2}$$

where c_{ji} is the resource of type j associated with a unit of type i.

One of the most convenient ways to solve this problem is reducing it to a one-dimensional problem. To this end, we introduce "weight" coefficients d_j such that: $0 \le d_j \le 1$, and

$$\sum_{1 \le j \le M} d_j = 1. \tag{10.3}$$

A set of d_j satisfying (10.3) presents a diagonal hyperplane within n-dimensional unitary hypercube. Denote this hyperplane by D. An explanation is given for a 3-dimensional case in the figure below.

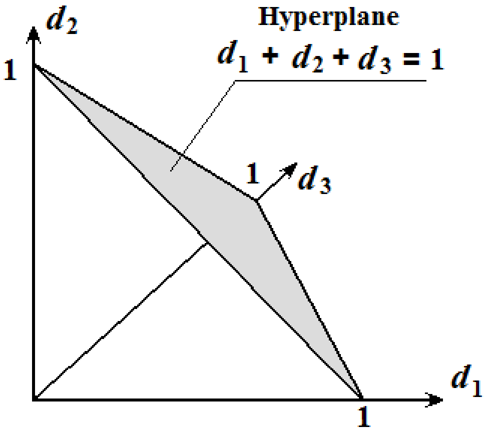


Figure 10.1. Hyperplaine with all possible d_j.

Use the Steepest Descent Method for the solution. The process of a solution is as follows. Choose a point $D^k = (d_1^k, d_2^k, \dots, d_M^k)$, $D^k \in D$. Produce for each unit j “weighed cost” c_j^k corresponding to vector D^k:

$$c_j^k = \sum_{1 \le k \le M} c_j d_j^k \tag{10.4}$$

Solving one-dimensional problem simultaneously controlling all M constrains. As soon as the optimization procedure has been stopped due to a possible violation of at least one of the constrains, the value of a reached level of R^k and realized vector X^k are memorized.

Then the next vector, say, D^j, is chosen and new values R^j and X^j are found. Compare admissible solutions and keep that with largest value of R.

The procedure of oriented choosing of D^k rather than direct enumerating can be organized: the procedure of the steepest descent could be used for this purpose.

Probably, even better procedure is as follows. At the stopping moment, pay attention to the type of constrain that is closest to violation. Sometimes it means that the process is "less optimal" relating to this type of resources. Increase a corresponding weight multiplier and repeat the process.

As in most practical cases, to find the appropriate choice of the increment to change d is more of an art than a science.

Maximum found value R corresponds to the optimal solution X^{opt}.

An illustrative example might be useful to demonstrate the method.

Numerical example.

Consider a series system consisting of three units with the characteristics given in Table 10.1.

Table 10.1 Data for example xxx.

Unit (i)	Reliability index P_i	Cost C_{i1}	Weight C_{i1}
1	0.7	3	1
2	0.8	5	1
3	0.9	2	3

A "hot" redundancy is permitted to improve the system reliability. The problem is to find the optimal solutions for the following constrains on the redundant system units as a whole:

(1) Cost: C_1^0= 15 conditional units; Weight: C_2^0 =15 conditional units;

(2) Cost: C_1^0= 20 conditional units; Weight: C_2^0 =15 conditional units.

<u>Solution.</u> Choose the increment for each d_i equal to 0.25. Then the "weighed cost" can be calculated as:

$$c_\Sigma^1 = c_i^{(1)}, \quad c_\Sigma^{0.75} = 0.75c_i^{(1)} + 0.25c_i^{(2)}, \quad c_\Sigma^{0.5} = 0.5c_i^{(1)} + 0.5c_i^{(2)},$$
$$c_\Sigma^{0.25} = 0.25c_i^{(1)} + 0.75c_i^{(2)}, \quad c_\Sigma^0 = c_i^{(2)}. \qquad (10.5)$$

Using (10.5), we get the following values:

Then one gets the following values for the equivalent costs:

c_1^2=2.5, c_2^2=4.0, c_3^2=2.25;
c_1^3=2.0, c_2^3=3.0, c_3^3=2.5;
c_1^4=1.5, c_2^4=2.0, c_3^4=2.75;
c_1^5=1.0, c_2^5=1.0, c_3^5=3.0 .

Now we separately solve all five problems for different equivalent costs. For simplicity, let us use the Steepest Descent Method. We omit all intermediate calculations that are routine and present only step-by-step results of the solution process.

Table 10.2. Solution process for various d_j.

d_k		1	2	3	4	5	6	7	8	9	10
1	X	1,0,0	1,0,1	1,1,1	2,1,1	2,2,1	3,2,1	3,2,2	4,2,2	4,3,2	5,3,2
	C_1	3	5	6	9	10	13	16	19	20	23
	C_2	1	4	9	10	15	16	17	18	23	24
0.75	X	0,1,0	1,1,0	1,2,0	2,2,0	3,2,0	3,3,0	4,3,0	4,3,1	5,3,1	6,3,1
	C_1	1	4	5	8	11	12	15	17	20	23
	C_2	5	6	11	12	13	18	19	22	23	24
0.5	X	0,1,0	1,1,0	1,2,0	2,2,0	3,2,0	4,2,0	5,2,0	5,2,1	6,2,1	6,3,1
	C_1	1	4	5	8	11	14	17	19	22	23
	C_2	5	6	11	12	13	14	15	18	19	24
0.25	X	1,0,0	1,1,0	2,1,0	2,2,0	3,2,0	4,2,0	5,2,0	5,3,0	6,3,0	7,3,0
	C_1	3	4	7	8	11	14	17	18	21	24
	C_2	1	6	7	12	13	14	15	20	21	22
0	X	1,0,0	1,1,0	2,1,0	3,1,0	3,2,0	4,2,0	5,2,0	6,2,0	6,3,0	7,3,0
	C_1	3	4	7	10	11	14	17	20	21	24
	C_2	1	6	7	8	13	14	15	16	21	20

Admissible solutions are (2,2,1) and (4,2,0). Solution (3,2,0) is not taken into account since it is dominated by (4,2,0). Let's now compare solutions:

$$R(2,2,1) = (1-0.3^3)(1-0.2^3)(1-0.1^2) = 0.9956$$

$$R(4,2,0) = (1-0.3^5)(1-0.2^3)\cdot 0.9 = 0.8906.$$

Thus the solution of the problem is vector (2,2,1), i.e. x_1=2, x_2=2, and x_3=1.

The inverse problem of optimal redundancy occurs in practice extremely rarely, so we omit its consideration.

10.2. Method of Generalized Generating Functions

The problem treated above can be solved exactly with the use of the Method of Generalized Generating Functions. The legion for each *i*th redundant group is represented as the set of the cohorts

$$L_i = \{ C_{i1}, C_{i2},\ldots,C_{iN_i}\}$$

where N_i is the number of cohorts in this legion. (In principle, the number of cohorts is unrestricted in this investigation.) Each cohort consists of M+2 maniples:

$$C_{ik} = (R_{ik}, c_{ik}^{1}, \ldots ,c_{ik}^{M}, x_{ik})$$

where M is the number of restrictions. All maniples are defined as in the one-dimension case that we considered above. A similar interaction is performed with the maniples:

$$\Omega^M_{c^k}(c^k_{ij_i}, c^k_{lj_l}) = c^k_{ij_i} + c^k_{l\,j_l} \quad and \quad \underset{1\le i\le n}{\Omega^M_{c^k}} c^k_i = \sum_{1\le i\le n} c^k_{ij_i}$$

$$\Omega^M_R(R_{ij}, R_{kl}) = R_{ij} R_{kl} \quad and \quad \underset{1\le i\le n}{\Omega^M_R} R_{i\,j_i} = \prod_{1\le i\le} R_{ij_i};$$

$$\Omega_x^M(x_{ij_i},\ x_{lj_l}) = (x_{ij_i},\ x_{lj_l}) \ \ \textit{and} \ \ \underset{1\le i\le n}{\Omega_x^M} x_{ij_i} = (x_{1j_1},\ x_{2j_2}, \dots ,\ x_{nj_n}) = \mathrm{X}_J$$

where **J** is the set of subscripts: $\boldsymbol{J}=(j_1, \dots ,j_n)$.

The remaining formal procedures totally coincide with the one-dimensional case with one very important exception: instead of a scaler ordering, one must use the special ordering of the cohorts of the final legion.

It is difficult to demonstrate the procedure on a numerical example, so we give only a detailed verbal explanation.

Suppose we have the file of current cohorts ordered according to increasing R. If we have a specified set of restrictions: $C_j(\mathbf{X})\leq C_{0j}$ for all j: $1\leq j\leq M]$, then there is no cohort in this file which violates at least one of these restrictions. When a new cohort, say, C_k, appears during the interaction procedure, it is put in the appropriate place in accordance with the value of its R-maniple. The computational problem is as follows.

1. Consider a part of the current file of cohorts for which the values of their R-maniple are less than the analogous value for C_k. If, among the existent cohorts, there is a cohort, say, C^*, for which all costs are larger than those of C_k, this cohort C^* is excluded from the file.

2. Consider a part of the current file of cohorts for which the values of the R-maniple are larger than the analogous value for C_k. If between the existent cohorts there is a cohort, say, C^{**}, for which all costs are smaller than those of C_k, the new cohort is not included in the file.

3. If neither 1 nor 2 take place, the new cohort is simply added to the file on the appropriate place.

After a multi-dimensional undominated sequence is constructed, one easily finds the solution for the multiple restrictions: it is the cohort with the largest R-maniple value (in other words, a cohort on the right if the set is ordered by the values of R).

The stopping rule for this procedure is to find the size of each cohort which will produce a large enough number of cohorts in the resulting legion so as to contain the optimal solution. At the same time, if the numbers of cohorts in the initial legions are too large, the computational procedure will take too much time and will demand too large a memory space.

Of course the simplicity of this description should not be deceptive. The problem is very bulky in the sense that the multi-dimensional restrictions and the large numbers of units in typical practical problems could require a huge memory and computational time. (But who can find a non-trivial multi-dimensional problem which has a simple solution?)

Bibliography to Chapter 10

1970. Proschan, F., and Bray, T.A. Optimum redundancy under multiple constraints. Operations Research, No. 13.

1971. Ushakov, I. A. Approximate solution of optimal redundancy problem for multipurpose system. *Soviet Journal of Computer and System Science,* No. 2.

1972. Ushakov, I. A. A heuristic method of optimization of the redundancy of multipurpose system. *Soviet Journal of Computer and System Sciences,* No. 4.

1981. Nakagawa, Y., and. Miyazaki, S. Surrogate constraints algorithm for reliability optimization problem with two constraints. IEEE Transaction on Reliability, No. 30.

1984. Genis, Ya. G., and Ushakov, I.A. Optimization of multi-purpose systems. *Soviet Journal of Computer and System Sciences*, No.3.

11.Optimal Redundancy in Multistate Systems

Solution of the problems of optimal redundancy allocation for multistate systems (MSS) consisting of multistate units is more laborious than solution of analogous problem for systems which have only two states: normal operation and failure.

Today this problem is investigatedin details. In first turn, the works by Gregory Levitin and Anatoly Lisnjanskij have to be mentioned (a complete nough lisy of hei papers is presented in the bibliography to the chapter.)

For more transparent explanation of the sense of the problem we begin with a simplest numerical example. Consider a series system of two different multistate units, each of which is characterized by several levels of performance. Performance may be measured by various physical values. Effectiveness of such system operation depends on levels of performance of Unit-1 and Unit-2.

Let units are characterized by the following parameters:

Table 11.1. Characterization of Unit-1

Level of performance (W_1)	**Probability p_1**	**Cost of a single unit**
100%	p_{11}=Pr{ W_1=100%}=0.9	c_1=1
70%	p_{12}=Pr{ W_1=100%}=0.05	
40%	p_{13}=Pr{ W_1=100%}=0.04	
0%	p_{14}=Pr{ W_1=100%}=0.01	

Table 11.2. Characterization of Unit-2

Level of performance (W_1)	**Probability p_2**	**Cost of a single unit**
100%	P_{21}=Pr{ W_2=100%}=0.8	c_2=2
80%	P_{22}=Pr{ W_2=80%}=0.18	
20%	P_{23}=Pr{ W_2=20%}=0.01	
0%	P_{24}=Pr{ W_2=0%}=0.01	

Assume that performance effectiveness of each unit can be improved by using loaded redundancy. Let suppose that at each moment of time, performance effectiveness of a redundant group is equal to the level of performance of the best component of the redundant group. Thus, behavior of Unit-1, consisting of the main component and single redundant element, can be depicted as in Figure 11.1.

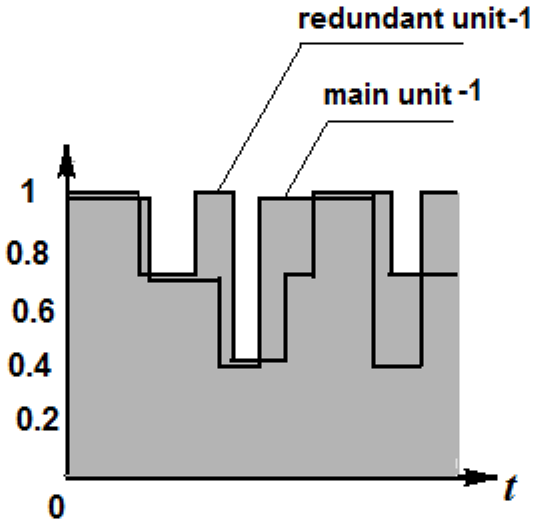


Figure 11.1. A realization of stochastic behavior of Unit-1, consisting of two elements, main and redundant. The shadowed area denotes the behavior of the Unit-1.

For Unit-2 analogous process is presented in Figure 11.2.

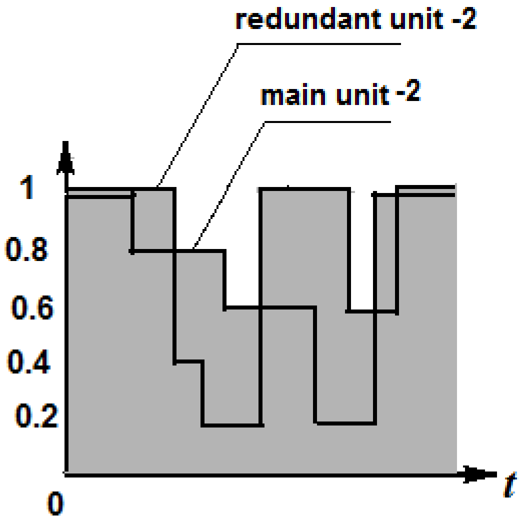


Figure 10.2. A realization of stochastic behavior of Unit-2, consisting of two elements, main and redundant. The shadowed area denotes the behavior of the Unit-2.

Further, assume that the entire system (series connection of Unit-1 and Unit-2) is characterized by the worst level of effectiveness of its units at each moment of time. In Figure 11.3, one can see the system behavior for the case when both units consist of a single main element.

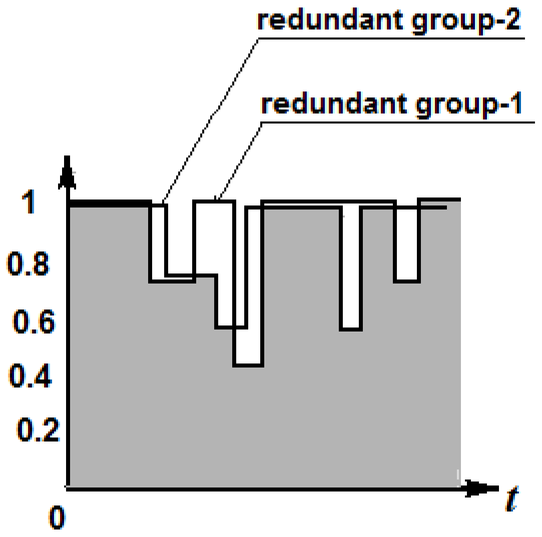


Figure 11.3. A realization of stochastic behavior of the entire system when both its units consist of a single main element. The shadowed area denotes the behavior of the system.

Let the problem is to find optimal redundant elements allocation described above series system:

(1) Direct problem: Find such an allocation of redundant elements than delivers average level of the system performance not less than required level of performance with minimum possible cost of redundant elements;
(2) Inverse problem: Find such an allocation of redundant elements than delivers maximum possible level of system performance under condition that the total expenses on redundant elements do not exceed the given total cost of redundant units.

Now consider construction of dominating sequence during the optimization process. In principle, one has to construct a table of type that presented below and choose members of dominating sequence.

Table 11.3. Construction of dominating sequence.

		Number of redundant elements for Unit-1			
		0	**1**	**2**	**…**

Number of redundant elements for Unit-2	**0**	X=(0, 0)
		P(0, 0)
		W(0,0)
		C(0, 0)
			X=(1, 0)
			P(1, 0)
			W(1, 0)
			C(1, 0)
				X=(2, 0)
				P(2, 0)
				W(2, 0)
				C(2, 0)
					…

	1	X=(0, 1)
		P(0, 1)
		W(0,1)
		C(0, 1)
			X=(1, 1)
			P(1, 1)
			W(1, 1)
			C(1, 1)
				X=(2, 1)
				P(2, 1)
				W(2, 1)
				C(2, 1)
					…

	2	X=(0, 2)
		P(0, 2)
		W(0, 2)
		C(0, 2)
			X=(1, 2)
			P(1, 2)
			W(1, 2)
			C(1, 2)
				X=(2, 2)
				P(2, 1)
				W(2, 2)
				C(2, 2)
					…

	…	…	…	…	…

As one sees, in this case we deal with quadruplets of type:
{[**Vector of numbers of redundant units];**
[Discrete distribution of performance levels];
[Performance levels];
[System cost]}.

The problem complicates due to necessity of calculations because “*Probabilities of performance levels*” and “*Performance levels*” are not numbers but vectors that needed special type of calculations. This aspect will be demonstrated below. Here we would like to note that there is no necessity to calculate quadruplets for all cells of Table 1. Fortunately, we can use the property of Kettelle Algorithm: members of dominating sequences are located around table’s diagonal and corresponding cells form simply connected area. It allows using “dichotomy tree” procedure, i.e. avoiding unnecessary calculations by cutting non-perspective branches (see Figure 11.4). Indeed, consider bordering cells around simple connected area (they marked with sign “x”.). There is no dominating cells in area located upper the right border, and there is no dominating cells in area located lower the left border.

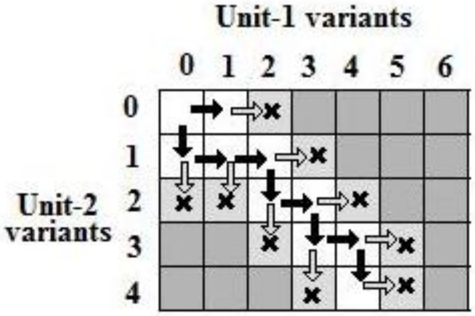


Figure 11.4. Example of excluding non-perspective branches. Black arrows are members of dominating sequence; grey ones are trial test that led to non-perspective variants marked by “x”. All cells marked with dark grey cannot contain dominating quadruplets.

Thus, in this case calculations occur to be sufficiently compact. However, as we mentioned above some special calculations for each redundant group have to be done.

In accordance with described above calculating procedure, one has to consider first variant (0, 0), i.e. just Unit-1 and Unit-2 with no redundancy at all, and find quadruple, In this case resulting solution will be:

$$\{[0]; [(p_{11}, W_{11}), (p_{12}, W_{12}), (p_{13}, W_{13}), (p_{14}, W_{14})]; [c]_1\} \otimes \{[0]; [(p_{21}, W_{21}), (p_{22}, W_{22}), (p_{23}, W_{23}), (p_{24}, W_{24})]; [c_2]\} =$$

$$\{0 \underset{\rightarrow}{\otimes} 0; [(p_{11}, W_{11}), (p_{12}, W_{12}), (p_{13}, W_{13}), (p_{14}, W_{14})]\} \underset{UGF}{\otimes} \{0; [(p_{21}, W_{21}), (p_{22}, W_{22}), (p_{23}, W_{23}), (p_{24}, W_{24})]; c_1 \underset{+}{\otimes} c_2\}. \qquad (11.1)$$

Here we use the following operators:

$\underset{\rightarrow}{\otimes}$ is an operator of forming a vector, i.e. $j \underset{\rightarrow}{\otimes} k = (j, k)$;

$\underset{UGF}{\otimes}$ is an operator equivalent to the U-function, i.e.

$\left(\sum_{j\in A} p_j z^{W_j}\right) \underset{UGF}{\otimes} \left(\sum_{k\in B} p_k z^{W_k}\right) = \sum_{\forall j,\forall k} p_j \cdot p_k z^{W_j \underset{\min}{\otimes} W_k}$, where, in turn, $W_j \underset{\min}{\otimes} W_k = \min(W_j, W_k)$;

$c_1 \underset{+}{\otimes} c_2$ is operator of summation, i.e. $c_1 \underset{+}{\otimes} c_2 = c_1 + c_2$.

Of course, the same procedure can be presented in terms of U-functions. One can write two “polynomials” of type

$$\left(\sum_{j\in A} p_{1j} x^0 y^{W_{1j}} z^{c_{1j}}\right) \underset{UGF}{\otimes} \left(\sum_{k\in B} p_{2k} x^0 y^{W_{2j}} z^{c_{2j}}\right) = \sum_{\forall j,\forall k} p_j \cdot p_k x^{0 \underset{\rightarrow}{\otimes} 0} y^{W_j \underset{\min}{\otimes} W_k} z^{c_{2j} \underset{+}{\otimes} c_{2k}} =$$

$$\sum_{\forall j,\forall k} p_j \cdot p_k x^{(0,0)} y^{\max(W_{1j}, W_{2k})} z^{c_{2j} \underset{+}{+} c_{2k}} .$$

(11.2)

Of course, a power of argument *x* has a very conditional sense: any value in “power” of vector has no common sense. For avoiding such confuses, we will operate with sequences of triplets, quadruplets and other “multiplets”.

Let us continue the numerical example because it helps us not explain relatively simple procedures on unnecessary formal level. Return to the series system, consisting of two units without redundancy. Numerical results are presented in Table 11.4.

Table 11.4. Initial state of the process of optimization.

(0, 0)		**Unit-2**					
		***p*₂₁=0.8** $W_{21}^{(0)}$ **=100%**	***p*₂₂=0.18** $W_{22}^{(0)}$ **=80%**			***p*₂₃=0.01** $W_{23}^{(0)}$ **=20%**	***p*₂₄=0.01** $W_{24}^{(0)}$ **=0%**
Unit-1	***p*₁₁=0.9** $W_{11}^{(0)}$ **=100%**	$p_{21} \cdot p_{11} = 0.72$ $\min(W_{21}^{(0)}, W_{11}^{(0)})$ =100%	$p_{22} \cdot p_{11} = 0.171$ $\min(W_{22}^{(0)}, W_{11}^{(0)})$ =80%			$p_{23} \cdot p_{14} = 0.009$ $\min(W_{23}^{(0)}, W_{11}^{(0)})$ =20%	$p_{24} \cdot p_{14} = 0.009$ $\min(W_{21}^{(0)}, W_{11}^{(0)})$ =0%
	***p*₁₂=0.05** $W_{12}^{(0)}$ **=70%**	$p_{21} \cdot p_{12} = 0.04$ $\min(W_{21}^{(0)}, W_{12}^{(0)})$ =70%	$p_{22} \cdot p_{12} = 0.0095$ $\min(W_{22}^{(0)}, W_{12}^{(0)})$ =70%			$p_{23} \cdot p_{14} = 0.0005$ $\min(W_{23}^{(0)}, W_{12}^{(0)})$ =20%	$p_{24} \cdot p_{14} = 0.0005$ $\min(W_{21}^{(0)}, W_{11}^{(0)})$ =0%
	***p*₁₃=0.04** $W_{13}^{(0)}$ **=40%**	$p_{21} \cdot p_{13} = 0.032$ $\min(W_{21}^{(0)}, W_{13}^{(0)})$ =40%	$p_{22} \cdot p_{13} = 0.0076$ $\min(W_{22}^{(0)}, W_{13}^{(0)})$ =40%			$p_{23} \cdot p_{14} = 0.0004$ $\min(W_{23}^{(0)}, W_{13}^{(0)})$ =20%	$p_{24} \cdot p_{14} = 0.0004$ $\min(W_{21}^{(0)}, W_{11}^{(0)})$ =0%
	***p*₁₄=0.01** $W_{14}^{(0)}$ **=0%**	$p_{21} \cdot p_{14} = 0.008$ $\min(W_{21}^{(0)}, W_{14}^{(0)})$ =0%	$p_{22} \cdot p_{14} = 0.0019$ $\min(W_{22}^{(0)}, W_{14}^{(0)})$ =0%			$p_{23} \cdot p_{14} = 0.0001$ $\min(W_{23}^{(0)}, W_{14}^{(0)})$ $= 0\%$	$p_{24} \cdot p_{14} = 0.0001$ $\min(W_{23}^{(0)}, W_{14}^{(0)})$ $= 0\%$

This leads to the following final result (see cells with the same background colors):
$P^{(0.0)}(W_{syst}=100\%)$ =0.72;
$P^{(0.0)}(W_{syst}=80\%)$ =0.171;
$P^{(0.0)}(W_{syst}=70\%)$ =0.04+0.0095=0.0495;
$P^{(0.0)}(W_{syst}=40\%$ =0.032+0.0076=0.0396;
$P^{(0.0)}(W_{syst}=20\%)$ =0.009+0.0005+0.004=0.0099;
$P^{(0.0)}(W_{syst}=0\%)$ =0.008+0.0019+0.0001+0.0001+0.0009+0.0005+0.0004= 0.0201.

Cost of additional units in this case equals 0. As one can easily calculate, the average level of the system performance is equal to

$$W_{syst}^{(0,0)} = 0.72 + 0.171 \cdot 0.8 + 0.0497 \cdot 0.7 + 0.0396 \cdot 0.5 + 0.0095 \cdot 0.2 \approx 0.9092.$$

Now let's make trial steps to the neighbor cells: check cells (1, 0) and (0, 1). Let us start with cell (1, 0) as it shown in Figure 11.4. First find the distribution of performance levels distribution for Unit-1 consisting of two elements, main and redundant.

Table 11.5. Forehand calculation of performance levels distribution for Unit-1, consisting of two elements, main and redundant.

		Element-1			
Element-1		***p_{11}*=0.9** $W_{11}^{(0)}$**=100%**	***p_{12}*=0.05** $W_{12}^{(0)}$**=70%**	***p_{13}*=0.04** $W_{13}^{(0)})$**=40%**	***p_{14}*=0.01** $W_{14}^{(0)}$**=0%**
	***p_{11}*=0.9** $W_{11}^{(0)}$**=100%**	$(p_{11})^2$=0.81 $W_{11}^{(0)}$=100%	$p_{12}\cdot p_{11}$=0.045 $\max(W_{12}^{(0)}, W_{11}^{(0)})$=100%	$p_{13}\cdot p_{11}$=0.036 max $(W_{13}^{(0)}, W_{11}^{(0)})$ =100%	$p_{14}\cdot p_{11}$=0.009 max $(W_{14}^{(0)}, W_{11}^{(0)})$ =100%
	***p_{12}*=0.05** $W_{12}^{(0)}$**=70%**	$p_{11}\cdot p_{12}$=0.045 max $(W_{11}^{(0)}, W_{12}^{(0)})$ =100%	$(p_{12})^2$=0.025 max$(W_{12}^{(0)}, W_{12}^{(0)})$=70%	$p_{13}\cdot p_{12}$=0.002 max $(W_{13}^{(0)}, W_{12}^{(0)})$ =70%	$p_{14}\cdot p_{12}$=0.0005 max $(W_{14}^{(0)}, W_{12}^{(0)}))$ =70%
	***p_{13}*=0.04** $W_{32}^{(0)}$**=40%**	$p_{11}\cdot p_{13}$=0.036 max $(W_{11}^{(0)}, W_{32}^{(0)})$ =100%	$p_{12}\cdot p_{13}$=0.002 max $(W_{12}^{(0)}, W_{32}^{(0)})$ =70%	$(p_{13})^2$=0.0016 $W_{32}^{(0)}$=40%	$p_{14}\cdot p_{13}$=0.0004 max $(W_{14}^{(0)}, W_{32}^{(0)})$ =40%
	***p_{14}*=0.01** $W_{14}^{(0)}$**=0%**	$p_{11}\cdot p_{14}$=0.009 max $(W_{11}^{(0)}, W_{14}^{(0)})$ =100%	$p_{12}\cdot p_{14}$=0.0005 max $(W_{12}^{(0)}, W_{14}^{(0)})$ =70%	$p_{13}\cdot p_{14}$=0.0004 max $(W_{13}^{(0)}, W_{14}^{(0)})$ =40%	$(p_{14})^2$=0.0001 $W_{14}^{(0)}$=0%

On the basis of this table, one gets for Unit-1 the following distribution

$\Pr\{W_1^{(1)} = 100\%\} = P_{11}^{(1)} = (p_{11})^2 + 2p_{11}\cdot(p_{12} + p_{13} + p_{14}) =$
$= 0.81 + 2\cdot(0.045+0.036+0.009) = 0.99;$
$\Pr\{W_1^{(1)} = 70\%\} = P_{12}^{(1)} = (p_{12})^2 + 2\cdot p_{12}\cdot(p_{13} + p_{14}) = 0.025 +$
$+ 2\cdot 0.025\,(0.002+0.0005) = 0.0075;$
$\Pr\{W_1^{(1)} = 40\%\} = P_{13}^{(1)} = (p_{13})^2 + 2p_{13}\cdot p_{14} = 0.0016 + 2\cdot 0.0016\cdot 0.0004 \approx 0.0016$:

$\Pr\{W_1^{(1)} = 0\%\} = P_{14}^{(1)} = (p_{14})^2$=0.0001.

Let us assume that the first step is made from (0, 0) to (1, 0) as it presented in Figure 11.5.

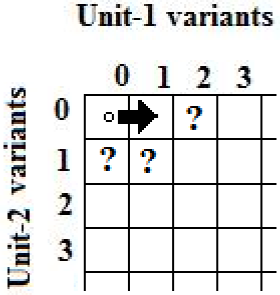


Figure 11.5. Direction of Step 1 of the optimization process.

Using the results, presented above, one can compile Table 11.6 that gives performance levels distribution for the system characterized by vector of redundant elements *X* = (1, 0).

Table 11.6. Step 1 of the optimization process.

C_{system}= (1, 0)		**Unit-2**					
		p_{21}=0.8 $W_{21}^{(0)}$)**=100%**	**p_{22}=0.19** $W_{22}^{(0)}$)**=80%**			**p_{23}=0.01** $W_{23}^{(0)}$)**=20%**	**p_{24}=0.01** $W_{24}^{(0)}$)**=0%**
Unit-1	**$P_{11}^{(1)}$=0.99** $W_{11}^{(1)}$ **=100%**	$p_{21} \cdot P_{11}^{(1)}$= 0.792 min $(W_{21}^{(0)}, W_{11}^{(1)})$ =100%	$p_{22} \cdot P_{11}^{(1)}$= 0.188 min$(W_{22}^{(0)}, W_{11}^{(1)})$ =80%			$p_{23} \cdot P_{11}^{(1)}$≈0.01 min$(W_{23}^{(0)}, W_{11}^{(1)})$ =20%	$p_{24} \cdot P_{11}^{(1)}$≈0.01 min$(W_{24}^{(0)}, W_{11}^{(1)})$) =0%
	=0.0075 $W_{12}^{(1)}$ **=70%**	$p_{21} \cdot P_{12}^{(1)}$= 0.006 min$(W_{21}^{(0)}, W_{12}^{(1)})$ =70%	$p_{22} \cdot P_{12}^{(1)}$≈0.0014 min$(W_{22}^{(0)}, W_{12}^{(1)})$ =70%			$p_{23} \cdot P_{12}^{(1)}$≈0.0001 min$(W_{23}^{(0)}, W_{12}^{(1)})$ =20%	$p_{24} \cdot P_{12}^{(1)}$=0.0001 min$(W_{24}^{(0)}, W_{12}^{(1)})$ =0%
	=0.0016 $W_{13}^{(1)}$ **=40%**	$p_{21} \cdot P_{13}^{(1)}$0.0013 min$(W_{21}^{(0)}, W_{13}^{(1)})$ =40%	$p_{22} \cdot P_{13}^{(1)}$≈0.0003 min$(W_{22}^{(0)}, W_{13}^{(1)})$ =40%			$p_{23} \cdot P_{13}^{(1)}$≈0 min$(W_{23}^{(0)}, W_{13}^{(1)})$ =20%	$p_{24} \cdot P_{13}^{(1)}$ ≈ 0 min$(W_{24}^{(0)}, W_{13}^{(1)})$ =0%
	=0.0001 $W_{14}^{(1)}$ **=0%**	$p_{21} \cdot P_{14}^{(1)}$≈0.0001 min$(W_{21}^{(0)}, W_{14}^{(1)})$ =0%	$p_{22} \cdot P_{14}^{(1)}$≈0 min$(W_{22}^{(0)}, W_{14}^{(1)})$ =0%			$p_{23} \cdot P_{14}^{(1)}$≈0 min$(W_{23}^{(0)}, W_{14}^{(1)})$ =0%	$p_{24} \cdot P_{14}^{(1)}$≈0 min$(W_{24}^{(0)}, W_{14}^{(1)})$ =0%

This leads to the following final result:

$P^{(1.0)}(W_{syst}=100\%)$ =0.792;
$P^{(0.0)}(W_{syst}=80\%)$ =0.188;
$P^{(0.0)}(W_{syst}=70\%)$ =0.006+0.0014=0.0074;
$P^{(0.0)}(W_{syst}=40\%$ =0.0013+0.0003=0.0016;
$P^{(0.0)}(W_{syst}=20\%)$ =0.01+0.0001=0.0101;
$P^{(0.0)}(W_{syst}=0\%)$ =0.008+0.0019+0.0001+0.0001+0.0009+0.0005+0.0004= 0.0201.

Cost of additional units in this case equals 1. Average system's performance level equals $W_{syst}^{(1,0)} = 0.792 + 0.188 \cdot 0.8 + 0.0497 \cdot 0.7 + 0.0396 \cdot 0.4 + 0.0095 \cdot 0.2 \approx 0.9502.$
Then try another neighbor cell, namely (0, 1). Beforehand, one has to perform an additional calculation of performance levels distribution for Unit-2 consisting of two elements, main and redundant.

It is necessary note that for parallel connection of multistate elements (that compiles a unit), more realistically to assume that the level of performance of the unit is equal to maximum among all currently operating elements. So, the table below represents results of calculation for Unit-2 that consists of two identical elements.

Table 11.7. Forehand calculation of performance levels distribution for Unit-2, consisting of two elements, main and redundant.

		Element-2			
Ele-ment -2		**p_{21}=0.8** $W_{21}^{(0)}$ **=100%**	**p_{22}=0.19** $W_{22}^{(0)}$ **=80%**	**p_{23}=0.01** $W_{23}^{(0)}$ **=20%**	**p_{24}=0.01** $W_{24}^{(0)}$ **=0%**
	p_{21}=0.8 $W_{21}^{(0)}$ **=100%**	$(p_{21})^2$=0.64 $W_{21}^{(0)}$ =100%	$p_{22} \cdot p_{21}$=0.045 $\max(W_{22}^{(0)}, W_{21}^{(0)})$ =100%	$p_{23} \cdot p_{21}$=0.036 $\max(W_{23}^{(0)}, W_{21}^{(0)})$ =100%	$p_{24} \cdot p_{21}$=0.008 $\max(W_{24}^{(0)}, W_{21}^{(0)})$ =100%
	p_{22}=0.19 $W_{21}^{(0)}$ **=80%**	$p_{21} \cdot p_{22}$=0.152 $\max(W_{21}^{(0)}, W_{21}^{(0)})$ =100%	$(p_{22})^2$=0.0361 $W_{22}^{(0)}$ =80%	$p_{23} \cdot p_{22}$=0.0002 $\max(W_{23}^{(0)}, W_{21}^{(0)})$ =80%	$p_{24} \cdot p_{22}$=0.0002 $\max(W_{24}^{(0)}, W_{21}^{(0)})$ =80%
	p_{23}=0.01 $W_{23}^{(0)}$ **=20%**	$p_{21} \cdot p_{23}$=0.008 $\max(W_{21}^{(0)}, W_{23}^{(0)})$ =100%	$p_{22} \cdot p_{23}$=0.0002 $\max(W_{22}^{(0)}, W_{23}^{(0)})$ =80%	$(p_{23})^2$=0.0001 $W_{23}^{(0)}$ =20%	$p_{24} \cdot p_{23}$=0.0001 $\max(W_{24}^{(0)}, W_{23}^{(0)})$ =20%
	p_{24}=0.01 $W_{24}^{(0)}$ **=0%**	$p_{21} \cdot p_{24}$=0.008 $\max(W_{21}^{(0)}, W_{24}^{(0)})$ =100%	$p_{222} \cdot p_{24}$=0.0002 $\max(W_{22}^{(0)}, W_{24}^{(0)})$ =70%	p $p_{23} \cdot p_{24}$=0.0001 $\max(W_{23}^{(0)}, W_{24}^{(0)})$ =20%	**$(p_{24})^2$=0.0001** $W_{24}^{(0)}$ **=0%**

On the basis of this table, one gets for Unit-2, consisting of two elements, the following distribution
$\Pr\{W_2^{(1)} = 100\%\} = P_{21}^{(1)}$= $(p_{21})^2$+2$p_{21}\cdot$ (p_{22}+ p_{23}+ p_{34}) = 0.64+2$\cdot$ 0.8$\cdot$ (0.045+0.036+0.008) ≈0.7709;
$\Pr\{W_2^{(1)} = 80\%\} = P_{22}^{(1)}$=$(p_{22})^2$+2 $\cdot p_{22} \cdot$ (p_{23}+ p_{24})=0.0361+2$\cdot$0.0361 (0.0002+0.0002) ≈ 0.0361;
$\Pr\{W_2^{(1)} = 20\%\} = P_{23}^{(1)}$=$(p_{13})^2$+2$p_{13}\cdot$ p_{14}=0.0001+0.0001+0.0001=0,0003:
$\Pr\{W_2^{(1)} = 0\%\} = P_{24}^{(1)} = (p_{14})^2$=0.0001.

After such preparations, one can make Step2 (see Figure 11.6)

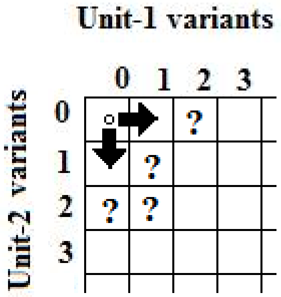


Figure 11.6. Direction of Step 2 of the optimization process.

This step consists in construction of Table 11.8 and presents the system's performance levels distribution for the system configuration characterized by vector of redundant elements *X* = (0, 1).

Table 11.8. Step 2 of the process of optimization.

C_{system}= (0, 1)		**Unit-2**					
		=0.7709 $W_{21}^{(1)}$ **=100%**	$P_{22}^{(1)}$**=0.0361** $W_{22}^{(1)}$ **=80%**			$P_{23}^{(1)}$**=0.0003** $W_{23}^{(1)}$ **=20%**	$P_{24}^{(1)}$**=0.0001** $W_{24}^{(1)}$ **=0%**
Unit-1	***p*₁₁=0.9** $W_{11}^{(0)}$ **=100%**	0.6038 min ($W_{21}^{(1)}, W_{11}^{(0)}$) =100%	$P_{22}^{(1)} \cdot p_{11} \approx 0.0325$ min($W_{22}^{(0)}, W_{11}^{(1)}$) =80%			$P_{23}^{(1)} \cdot p_{11} \approx 0.0003$ min($W_{23}^{(0)}, W_{11}^{(1)}$) =20%	$P_{24}^{(1)} \cdot p_{11} \approx 0.0001$ min($W_{24}^{(0)}, W_{11}^{(1)}$)) =0%
	***p*₁₂=0.05** $W_{12}^{(0)}$ **=70%**	0.0386 min($W_{21}^{(0)}, W_{12}^{(1)}$) =70%	$P_{22}^{(1)} \cdot p_{12} \approx 0.0018$ min($W_{22}^{(0)}, W_{12}^{(1)}$) =70%			$P_{23}^{(1)} \cdot p_{12} \approx 0$ min($W_{23}^{(0)}, W_{12}^{(1)}$) =20%	$P_{24}^{(1)} \cdot p_{12} \approx 0$ min($W_{24}^{(0)}, W_{12}^{(1)}$) =0%
	***p*₁₃=0.04** $W_{32}^{(0)}$ **=40%**	0.0308 min($W_{21}^{(0)}, W_{13}^{(1)}$) =40%	$P_{22}^{(1)} \cdot p_{13} \approx 0.0014$ min($W_{22}^{(0)}, W_{13}^{(1)}$) =40%			$P_{23}^{(1)} \cdot p_{13} \approx 0$ min($W_{23}^{(0)}, W_{13}^{(1)}$) =20%	$P_{24}^{(1)} \cdot p_{13} \approx 0$ min($W_{24}^{(0)}, W_{13}^{(1)}$) =0%
	***p*₁₄=0.01** $W_{14}^{(0)}$ **=0%**	≈0.00771 min($W_{21}^{(0)}, W_{14}^{(1)}$) =0%	$P_{22}^{(1)} \cdot p_{14} \approx 0.0004$ min($W_{22}^{(0)}, W_{14}^{(1)}$) =0%			$P_{23}^{(1)} \cdot p_{14} \approx 0$ min($W_{23}^{(0)}, W_{14}^{(1)}$) =0%	$P_{24}^{(1)} \cdot p_{14} \approx 0$ min($W_{24}^{(0)}, W_{14}^{(1)}$) =0%

This leads to the following final result:
$P^{(1.0)}(W_{syst}=100\%) = 0.6038$;
$P^{(0.0)}(W_{syst}=80\%) = 0.0325$;
$P^{(0.0)}(W_{syst}=70\%) = 0.0386+0.0018=0.0404$;

$P^{(0.0)}(W_{syst}=40\% =0.0308+0.0014=0.0322;$
$P^{(0.0)}(W_{syst}=20\%) \approx 0.0003;$
$P^{(0.0)}(W_{syst}=0\%) =0.0077+0.0004+0.0001 \approx 0.0082.$

Cost of additional units in this case equals 2 units of cost. Average system's performance level equals

$$W_{syst}^{(0,1)} = 0.6038 + 0.0.0325 \cdot 0.8 + 0.0404 \cdot 0.7 + 0.0322 \cdot 0.4 + 0.0003 \cdot 0.2 \approx 0.671.$$

Thus, for vector (1, 0) one has additional cost equal 1 and $W_{syst}^{(1,0)} \approx 0.9502$ and for vector (0, 1) corresponding values equal to 2 and 0.671, so system configuration (1, 0) is dominating over configuration (0, 1), since higher average performance level delivers with less expenses. It means that all vectors of type (0, *k*) are excluded from further analysis.

The next cells, for which current trials have to be done, are cells (1, 1) and (2, 0).

Avoiding simple, however cumbersome calculations, let us present only final results (see Table 11.9).

Table 11.9. Costs and levels of performance for different vectors of redundant elements.

		Unit-1: Number of redundant elements						
		0	1	2	3	4	5	6
Unit-2: Number of redundant elements	0	C=0 W= 90.16	C=1 W= 94.26	C=2 W= 94.57				…
	1	C=2 W= 94.68	C=3 W= 99.16	C=4 W= 99.50	C=5 W= 99.53			…
	2	C=4 W= 95.03	C=5 W= 99.54	C=6 W= 99.89	C=7 W= 99.92	**?**		…
	3		C=7 W= 99.61	C=8 W= 99.95	**?**			… .
	4			**?**				… .
	…	…	… .	…	…	…	…	…
	Legend: light grey color – dominated cells, dark grey color – non-prospective variants.							

The table above is constructed as it shown in Figure 11.7.
\

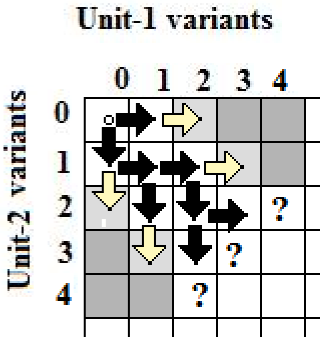


Figure 117. The process of step-by-step development of the optimization process.

Probably, the last table needs some explanations.. Vector (2, 0) is dominated by vector (0, 1), since vector (0, 1) is characterized by higher performance level for the same total cost of redundant units. So, all vectors of type (3, 0),(4, 0), …, (*k*, 0), … are excluded from the further consideration. The same type of domination one observes for the following pairs: (3, 1) is dominated by (1, 2), vector (0, 2) is dominated by (1, 1), vector (1, 3) is dominated by (2, 2) and so on.

Such trials and selection of dominating vectors continued until appearance of first vector with the average level of performance higher than required value of W^o for the direct problem of optimal redundancy, or until total expense of all redundant elements are not exceed given value C^o for the inverse problem. These comments become absolutely transparent if one takes a look on Figure 118.

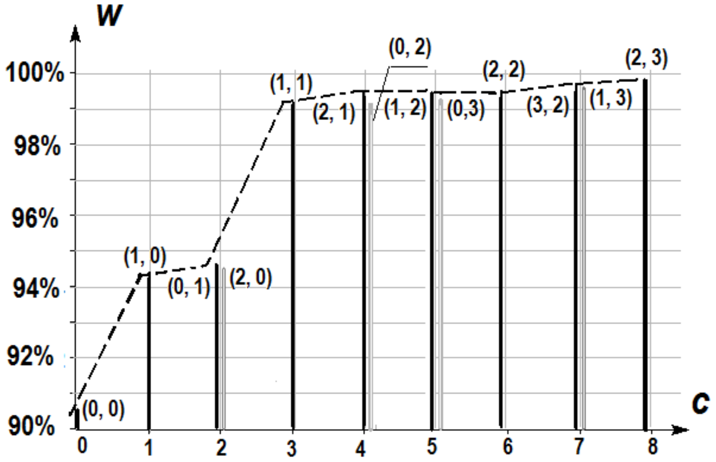


Figure 11.. Depiction of the process of compiling the dominating sequence.

From Table 11.9, one can see that optimal solution for requirement that the average level of system performance is not less than W^o =0.999 is delivered by vector (3, 2), and the total expenses of redundant elements is 7 cost units. For the total expenses on redundant elements limited by $C^o \leq 5$ cost units, one gets maximum possible solution as vector (1, 2) that characterizes by W=99.54%.

It is interesting what happens with the optimal solution if one changes costs of elements> Let us assume that for the same system cost of a single redundant element of the 1st type is c_1=2 and the cost am element of the 2nd type c_2=1.

Table 11.10. Costs and levels of performance for different vectors of redundant elements for new element's costs.

		Unit-1: Number of redundant elements						
		0	1	2	3	4	5	6
	0	C=0 W=90.156	C=2 W= 94.26456	▨	▨	▨	▨	…
Unit-2: Number of redundant elements	1	C=1 W= 94.68072	C=3 W= 99.16318	C=5 W= 99.50462	▨	▨	▨	…
	2	C=2 W= 95.02683	C=4 W= 99.54058	C=6 W= 99.88507	C=8 W= 99.9156	▨	▨	…
	3	C=3 W= 95.08558	C=5 W= 99.60514	C=7 W= 99.9502	**?**			… .
	4	▨	C=6 W= 99.61784	**?**				… .
	…	…	… .	…	…	…	…	…
	Legend: light grey color – dominated cells, dark grey color – non-prospective variants.							

In this case optimal solutions found from Table 11.10 are: For the direct problem vector (2, 3), for which W=99.95% and total expenses on redundant elements are equal to 7 cost units, and for inverse problem the solution is (1, 3), for which W=99.54% and total expenses C=5.

Solution of optimal redundancy problems for system consisting of several multilevel units seems a bit cumbersome. However, let us note that all enumerative methods like dynamic programming practically unsolvable without computerizing calculations. Numerical example above was solved with the help of a simple programs using Microsoft Excel.

For complex systems consisting of *n* multiple multistate units, one can compile a simple program for a mainframe computer. The algorithm should include the following steps.

1. Take an *n*-dimensional vector of redundant elements $X^{(0)} = (x_1^{(0)} = 0, x_2^{(0)} = 0, ..., x_n^{(0)} = 0)$.
2. Perform calculations to get initial pair of values $(W_{syst}^{(0)}, C_{syst}^{(0)})$ (see Table 11.2).
3. Put calculated pair $(W_{syst}^{(0)}, C_{syst}^{(0)})$ into list of dominating solutions,

4. Generate vectors $X_i^{(1)}$ such that each of them distinguishes from $X^{(0)}$ by changing number of elements of Unit-*i* on one, i.e. $X_i^{(1)} = (x_1^{(0)} = 0, x_2^{(0)} = 0, ..., x_i^{(0)} = 1, ..., x_n^{(0)} = 0)$.
5. For each $X_i^{(1)}$, $\overline{i=1,n}$, calculate new values of $P_{ik_i}^{(1)}$, for all k_i where k_i is the number of performance levels of Unit-*I*.
6. Perform calculations to get *n* pairs $(W_1^{(1)}, C_1^{(1)}), (W_2^{(1)}, C_2^{(1)}), ..., (W_n^{(1)}, C_n^{(1)}),$ for all vectors.

Such solution appears a bit clumsy and laborious. However, computer calculating program is relatively simple and solution can be obtained easy enough; final results are presented in the form of dominating sequence (in Kettelle's terminology),so solution for direct and/or inverse problem optimal redundancy can be easily found..

Conclusion.We restrict ourselves by consideration this simple and ore or less transparent illustrative example. Last years this problematic generates a number of interesting and theoretically deep publications, as the reader can see from bibliography below, However, we think that more detailed consideration of this problem could lead us too far from the "highway" of main practical optimal redundancy tasks.

Bibliography to Chapter 11

(books are distinguished by bold font)

1985. Reinschke, K. Systems Consisting of Units with Multiple States. In Handbook: Reliability of Technical Systems (Ed. I. Ushakov). Sovetskoe Radio, Moscow..

1988. El-Neweihi, E., Proschan, F., and Sethuraman, J. Optimal allocation of multistate components. In: *Handbook of Statistics, Vol.7: Quality Control and Reliability*. Edited by P.R.Krishnaiah, C.R.Rao. North-Holland.

1988. Ushakov, I. Reliability analysis of multi-state systems by means of modified generating function. Elektronische Informationsverarbeitung und Kybernetik, No. 3.

1998. Levitin, G., and Lisnianski, A. Joint redundancy and maintenance optimization for multistate series-parallel systems. Reliability Engineering & System Safety, No.64, No.1.

1998. Levitin, G., Lisnianski, A., Ben Haim, H., and Elmakis, D. Redundancy optimization for series-parallel multi-state systems, IEEE Transactions on Reliability, , No. 2.

1999. Levitin, G., and Lisnianski, A. Importance and sensitivity analysis of multi-state systems using universal generating functions method, Reliability Engineering & System Safety, No. 65.

2000. Levitin, G., and Lisnianski, A. Optimal Replacement Scheduling in Multi-state

Series-parallel Systems. Quality and Reliability Engineering International, No. 16.

2000. Levitin, G., Lisnianski, A., and Ben Haim, H. Structure optimization of multi-state system with time redundancy. Reliability Engineering and System Safety, No. 67.

2001. Levitin, G. Redundancy optimization for multi-state system with fixed resource-requirements and unreliable sources. IEEE Transactions on Reliability, No. 50.

2001. Levitin, G., and Lisnianski, A. A new approach to solving problems of multi-state system reliability optimization, Quality and Reliability Engineering International, No. 47.

2001. Levitin, G., and Lisnianski, A. Structure Optimization of Multi-state System with two Failure modes. Reliability Engineering and System Safety, No. 72.

2002. Levitin, G. Optimal allocation of multi-state elements in linear consecutively-connected systems with delays. International Journal of Reliability Quality and Safety Engineering, No. 9.

2002. Levitin, G., Lisnianski, A., and Ushakov, I. Multi-state system reliability: from theory to practice, with G.. *Proc. of the 3rd Internationall Conference on Mathematical Models in Reliability,* Trondheim, Norway.

2003. Levitin, G. Optimal multilevel protection in series-parallel systems, Reliability Engineering & System Safety, No. 81.

2003. Levitin, G. Optimal allocation of multi-state elements in linear consecutively-connected systems. IEEE Transactions on Reliability, No. 2.

2003. Levitin, G., Dai, Y., Xie, M., and Poh, K. L. Optimizing survivability of multi-state systems with multi-level protection by multi-processor genetic algorithm. Reliability Engineering & System Safety, No. 82.

2003. Levitin,G., Lisnianski, A., and Ushakov, I. Reliability of Multi-state Systems: A Historical Overview. Mathematical and Statistical Methods in Reliability, No. 7.

2003. Lisnianski, A., and Levitin, G. Multi-state System Reliability: Assessment, Optimization and Applications, World Scientific.

2004. Levitin, G. A universal generating function approach for analysis of multi-state systems with dependent elements. Reliability Engineering & System Safety, No.3.

2004. Nourelfath, M., and Dutuit, Y. A combined approach to solve the redundancy optimization problem for multi-state systems under repair policies, Reliability Engineering & System Safety, No.3.

2004. Ramirez-Marquez, J.E., and Coit, D.W. A heuristic for solving the redundancy allocation problem for multi-state series-parallel systems. Reliability Engineering & System Safety, No. 83.

2008. Ding, Y., and Lisnianski, A. (2008) Fuzzy universal generating functions for multi-state system reliability assessment. Fuzzy Sets and Systems, No. 3.

2008. Tian.Z, Zuo, M.J., and Huang, H. Reliability-Redundancy Allocation for Multi-State Series-Parallel Systems. IEEE Transactions on Reliability, No. 2.